A MEGERŐSÍTÉSES TANULÁS MÓDSZEREI

Jakab Hunor

2015. április 22.

Magyar Matematika és Informatika Intézet
Babes-Bolyai Tudományegyetem

BEVEZETÉS

MEGERŐSÍTÉSES TANULÁS - KUTATÁSI IRÁNYVONALAK

Jellemzők

- Optimális döntéshozatali stratégiák tanulása

MEGERŐSÍTÉSES TANULÁS - KUTATÁSI IRÁNYVONALAK

Jellemzők

- Optimális döntéshozatali stratégiák tanulása
- Nincs külső beavatkozás

MEGERŐSÍTÉSES TANULÁS - KUTATÁSI IRÁNYVONALAK

Jellemzők

- Optimális döntéshozatali stratégiák tanulása
- Nincs külső beavatkozás
- Nincs tanuló adathalmaz

MEGERŐSÍTÉSES TANULÁS - KUTATÁSI IRÁNYVONALAK

Jellemzők

- Optimális döntéshozatali stratégiák tanulása
- Nincs külső beavatkozás
- Nincs tanuló adathalmaz
- Feladat specifikus jutalmi függvény *reward signal*

MEGERŐSÍTÉSES TANULÁS - KUTATÁSI IRÁNYVONALAK

Jellemzők

- Optimális döntéshozatali stratégiák tanulása
- Nincs külső beavatkozás
- Nincs tanuló adathalmaz
- Feladat specifikus jutalmi függvény *reward signal*

MEGERŐSÍTÉSES TANULÁS - KUTATÁSI IRÁNYVONALAK

Jellemzők

- Optimális döntéshozatali stratégiák tanulása
- Nincs külső beavatkozás
- Nincs tanuló adathalmaz
- Feladat specifikus jutalmi függvény *reward signal*

Valós világ-beli feladatok tulajdonságai

- Folytonos, több dimenziós állapotok $\rightarrow$ végtelen állapottér
- Folytonos végrehajtható műveletek $\rightarrow$ végtelen művelettér
- Közelítések alkalmazása elkerülhetetlen
- A rendszer nem indítható tetszőleges állapotból
- A végrehajtható kísérletek száma korlátozott

JELÖLÉSEK

Markov döntési folyamatok

$M\,(S, A, P, R)$:

- S - állapotok halmaza
- A - műveletek halmaza
- P - állapot átmenetek valószínűsége:
  $P(s'|s, a) : S \times S \times A \to [0, 1]$
- $R(s, a) : S \times A \to R$ jutalom függvény

JELÖLÉSEK

Markov döntési folyamatok

$M\,(S, A, P, R)$:

- S - állapotok halmaza
- A - műveletek halmaza
- P - ismeretlen állapot átmenetek valószínűsége: $P(s'|s, a) : S \times S \times A \to [0, 1]$
- $R(s, a) : S \times A \to R$ jutalom függvény

JELÖLÉSEK

Markov döntési folyamatok

$M\,(S, A, P, R)$:

- S - állapotok halmaza
- A - műveletek halmaza
- P - ismeretlen állapot átmenetek valószínűsége: $P(s'|s, a) : S \times S \times A \to [0, 1]$
- $R(s, a) : S \times A \to R$ jutalom függvény

Stratégia (*policy*)

$\pi : S \times A \to [0, 1]$

- $\pi(a|s)$ - műveletek választásának feltételes valószínűsége

Cél:

Optimális stratégia keresése:

- $\pi^* = \arg\max_\pi E_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t\right]$

Értékelő függvények

Állapotok vagy állapot-művelet párok hasznosságának a becslésére szolgálnak:

- $V_\pi(s) = E_\pi\left[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s\right]$
- $Q_\pi(s,a) = E_\pi\left[\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s, a_0 = a\right]$

Értékelő függvények közelítése

Tanuló adatok

- Nem állnak előre rendlekezésünkre

Értékelő függvények közelítése

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**
  - s_t - kiinduló állapot

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**
  - s_t - kiinduló állapot
  - a_t - végrehajtott művelet

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**
  - s_t - kiinduló állapot
  - a_t - végrehajtott művelet
  - r_t - jutalmi függvény által visszatérített érték

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**
  - s_t - kiinduló állapot
  - a_t - végrehajtott művelet
  - r_t - jutalmi függvény által visszatérített érték
  - s'_t - a következő állapot amelybe az MDP átlép :

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**
  - s_t - kiinduló állapot
  - a_t - végrehajtott művelet
  - r_t - jutalmi függvény által visszatérített érték
  - s'_t - a következő állapot amelybe az MDP átlép :

$$s'_t = \int_S \int_A P(s|s_t, a_t)\pi(a|s_t)\, da\, ds$$

ÉRTÉKELŐ FÜGGVÉNYEK KÖZELÍTÉSE

Tanuló adatok

- Nem állnak előre rendlekezésünkre
- A környezettel való interakció során kapjuk meg őket (mérések, jutalmi függvény)
- Egy művelet végrehajtása egy újabb mintát eredményez: $x_t = \{s_t, a_t, r_t, s'_t\}$
  - t - a lépés sorszáma **(idő)**
  - s_t - kiinduló állapot
  - a_t - végrehajtott művelet
  - r_t - jutalmi függvény által visszatérített érték
  - s'_t - a következő állapot amelybe az MDP átlép :

$$s'_t = \int_S \int_A P(s|s_t, a_t)\pi(a|s_t)\, da\, ds$$

KÖZELÍTÉSI LEHETŐSÉGEK

- **parametrikus függvényközelítők** (*lineáris architektúrák nemlineáris feature függvényekkel, neuron hálók, stb.*)
- **nemparametrikus függvényközelítők** (*Gauss folyamatok, döntési fák stb.*)

KÖZELÍTÉSI LEHETŐSÉGEK

- **parametrikus függvényközelítők** (*lineáris architektúrák nemlineáris feature függvényekkel, neuron hálók, stb.*)
- **nemparametrikus függvényközelítők** (*Gauss folyamatok, döntési fák stb.*)

KRLSTD

Próbálkozás: *Kernel Recursive Least Squares Temporal Difference Learning*

- lineáris architektúra nemparametrikus megközelítésben
- időbeli differenciák felhasználása
- paraméterek számának korlátozása
- on-line módszer

KRLSTD

Lineáris parametrikus függvényközelítőből indulunk ki

$\tilde{V}(s) = \phi(s)^T \mathbf{w}$

- $\phi(s) = [\phi_1(s), \ldots, \phi_d(s)]^T$
- $\mathbf{w} \in \mathbb{R}^d$ súlyok
- d - a feature tér dimenzionalitása

A függvényközelítőt lépésenként frissítjük az időbeli differenciák alapján

KRLSTD

A függvényközelítőt lépésenként frissítjük az időbeli differenciák alapján

$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \alpha_t \delta_t \phi(s_t)$.

- $\delta t = R_t - (\phi(s_t) - \gamma\phi(s'_t))^T \mathbf{w}_t$

konvergencia

Végtelen számú lépés esetén a $\mathbf{w}$ súlyok értékei konvergálnak:

$$E[\phi(s_t)\delta_t] = 0$$

Az időbeli differenciák várható értékben 0-hoz tartanak

KRLSTD

A várható értéket helyettesítjük:

$\frac{1}{n} \sum_{t=0}^{n} \phi(s_t)\delta_t = 0$

KRLSTD

A várható értéket helyettesítjük:

$\frac{1}{n} \sum_{t=0}^{n} \phi(s_t)\delta_t = 0$

Kifejtve a fenti egyenletet a következőkhöz jutunk:
$\left(\sum_{t=0}^{n} \phi(s_t) \left(\phi(s_t) - \gamma\phi(s'_t)\right)^T\right) \mathbf{w} = \sum_{t=0}^{n}, \phi(s_t)R_t$

KRLSTD

A várható értéket helyettesítjük:

$\frac{1}{n}\sum_{t=0}^{n}\phi(s_t)\delta_t = 0$

Kifejtve a fenti egyenletet a következőkhöz jutunk:
$\left(\sum_{t=0}^{n}\phi(s_t)\left(\phi(s_t)-\gamma\phi(s_t')\right)^T\right)\mathbf{w} = \sum_{t=0}^{n},\phi(s_t)R_t$
Bevezetjük a következő jelölést:

$$\tilde{A} = \sum_{t=0}^{n}\phi(s_t)\left(\phi(s_t)-\gamma\phi(s_t')\right)^T, \tag{1}$$

$$\tilde{b} = \sum_{t=0}^{n}\phi(s_t)R_t, \tag{2}$$

Egyszerűsített alak

$$\tilde{A}\mathbf{w} = \tilde{b}$$

Tulajdonságok

- Az $\tilde{A}$ mátrix külső szorzatok összege
- Az összeg minden tagja egy négyzetes mátrix
- Amennyiben $\tilde{A}$ invertálható, van azonnali megoldásunk $\mathbf{w}$ -re
- Nemparaméteres formára hozható kernel függvények bevezetésével

RKHS

- Feltételezzük, hogy a vetítés egy RKSH-be történik
- A megoldás felírható úgy mint a feature vektorok lineáris kombinációja: $\mathbf{w} \overset{\text{def}}{=} \sum_{t=1}^{n} w_i \phi(s_i)$ ahol n a megfigyelt training adatok száma

Visszahelyettesítve az előző egyenletünkbe:
$\left(\sum_{i=0}^{n} \phi(s_t) \left(\phi(s_t) - \gamma\phi(s'_t)\right)^T\right) \mathbf{w} = \sum_{t=0}^{n}, \phi(s_t) R_t$ a következő rendszerhez jutunk: $\mathbf{w}^* = \tilde{A}^{-1}\tilde{\mathbf{b}}$

$$\tilde{A} = \sum_{t=0}^{n} \mathbf{k}(s_t) \left[\mathbf{k}^T(s_t) - \gamma \mathbf{k}^T(s'_t)\right], \tag{3}$$

$$\tilde{\mathbf{b}} = \sum_{t=0}^{n} \mathbf{k}(s_t)\, R_t, \tag{4}$$

KERNEL FÜGGVÉNYEK HASZNÁLATA

Megjegyzések:

- A belső szorzatokat $\phi(s_i)^T \phi(s_j) \rightarrow k(s_i, s_j)$ kernel függvényekkel helyettesítettük.

KERNEL FÜGGVÉNYEK HASZNÁLATA

Megjegyzések:

- A belső szorzatokat $\phi(s_i)^T\phi(s_j) \to k(s_i, s_j)$ kernel függvényekkel helyettesítettük.
- Egy adott s ponthoz tartozó kernel értékeket tartalmazó vektor $\mathbf{k}(s) = [k(s, s_1), \ldots, k(s, s_n)]^T$ ahol a tréning adatok halmaza: $D_n = \{s_i \,|i = \overline{1, n}\}$

KERNEL FÜGGVÉNYEK HASZNÁLATA

Megjegyzések:

- A belső szorzatokat $\phi(s_i)^T\phi(s_j) \to k(s_i, s_j)$ kernel függvényekkel helyettesítettük.
- Egy adott s ponthoz tartozó kernel értékeket tartalmazó vektor $\mathbf{k}(s) = [k(s, s_1), \ldots, k(s, s_n)]^T$ ahol a tréning adatok halmaza: $D_n = \{s_i \,|i = \overline{1, n}\}$
- Következmény: $\tilde{V}(s) = \sum_{i=0}^{n} w_i^* k(s, s_i)$ ahol $\mathbf{w}^* = \tilde{A}^{-1}\tilde{\mathbf{b}}$

KERNEL FÜGGVÉNYEK HASZNÁLATA

Megjegyzések:

- A belső szorzatokat $\phi(s_i)^T\phi(s_j) \to k(s_i, s_j)$ kernel függvényekkel helyettesítettük.
- Egy adott s ponthoz tartozó kernel értékeket tartalmazó vektor $\mathbf{k}(s) = [k(s, s_1), \ldots, k(s, s_n)]^T$ ahol a tréning adatok halmaza: $D_n = \{s_i \,|i = \overline{1, n}\}$
- Következmény: $\tilde{V}(s) = \sum_{i=0}^{n} w_i^* k(s, s_i)$ ahol $\mathbf{w}^* = \tilde{A}^{-1}\tilde{\mathbf{b}}$

KERNEL FÜGGVÉNYEK HASZNÁLATA

Megjegyzések:

- A belső szorzatokat $\phi(s_i)^T\phi(s_j) \to k(s_i, s_j)$ kernel függvényekkel helyettesítettük.
- Egy adott s ponthoz tartozó kernel értékeket tartalmazó vektor $\mathbf{k}(s) = [k(s, s_1), \ldots, k(s, s_n)]^T$ ahol a tréning adatok halmaza: $D_n = \{s_i \,|i = \overline{1, n}\}$
- Következmény: $\tilde{V}(s) = \sum_{i=0}^{n} w_i^* k(s, s_i)$ ahol $\mathbf{w}^* = \tilde{A}^{-1}\tilde{\mathbf{b}}$

Javítási lehetőségek:

- Az köbös költségű mátrix inverzió elkerülése
- Iteratív kiszámolása a $\tilde{A}^{-1}$ és a $\tilde{\mathbf{b}}$-nek.

ITERATÍV FRISSÍTÉSEK

ITERATÍV UPDATE

A $\tilde{A}$ mátrix bővítése

$$\tilde{A}_{t+1} \quad \stackrel{\text{def}}{=} \quad \frac{t}{t+1}\left(\begin{bmatrix}\tilde{A}_t & \mathbf{0} \\ \mathbf{0}^T & 0\end{bmatrix} + \begin{bmatrix}\mathbf{u} \\ u^*\end{bmatrix}\begin{bmatrix}\mathbf{v} \\ v^*\end{bmatrix}^T\right)$$

Ahol a következő jelöléseket használtuk:

$$\mathbf{u} = \begin{bmatrix}k(s_{t+1}, s_1) \\ \vdots \\ k(s_{t+1}, s_t)\end{bmatrix}, \mathbf{v} = \begin{bmatrix}k(s_{t+1}, s_1) - \gamma k(s'_{t+1}, s_1) \\ \vdots \\ k(s_{t+1}, s_t) - \gamma k(s'_{t+1}, s_t)\end{bmatrix}$$

.

$$u^* = k(s_{t+1}, s_{t+1}) \quad \text{és} \quad v^* = k(s_{t+1}, s_{t+1}) - \gamma k(s'_{t+1}, s_{t+1})$$

ITERATÍV UPDATE

Legyen: $C_t = \tilde{A}_t^{-1}$
Az inverz iteratív kiszámolása a következőképpen történik:

$$C_{t+1} = \tilde{A}_{t+1}^{-1} = \frac{t+1}{t} \begin{bmatrix} \tilde{A}_t + \mathbf{u}\mathbf{v}^T & v^*\mathbf{u} \\ u^*\mathbf{v}^T & u^*v^* \end{bmatrix}^{-1} \tag{5}$$

A Sherman Woodbury képlet többszörös használata után a következő, viszonylag egyszerű alakra jutunk:

$$C_{t+1} = \begin{bmatrix} C_t & -C_t\mathbf{u}/u^* \\ -\mathbf{v}^T C_t/v^* & \frac{1+\mathbf{v}^T C_t \mathbf{u}}{u^* v^*} \end{bmatrix} \tag{6}$$

ITERATÍV UPDATE

A $\tilde{\mathbf{b}}$ frissítése:

$$\tilde{\mathbf{b}}_{t+1} = \frac{t}{t+1}\left(\begin{bmatrix}\tilde{\mathbf{b}}_t \\ 0\end{bmatrix} + \begin{bmatrix}\mathbf{u} \\ u^*\end{bmatrix} R_t\right) \tag{7}$$

A $\tilde{A}^{-1}$ és $\tilde{\mathbf{b}}$ iteratív frissítéseit kombinálva kifejezhető a végső megoldás:

$$\begin{aligned}
\mathbf{w}_{t+1} &= C_{t+1}\tilde{\mathbf{b}}_{t+1} \\
&= \begin{bmatrix} C_t & -C_t\mathbf{u}/u^* \\ -\mathbf{v}^T C_t/v^* & \frac{1+\mathbf{v}^T C_t \mathbf{u}}{u^* v^*} \end{bmatrix}\left(\begin{bmatrix}\tilde{\mathbf{b}}_t \\ 0\end{bmatrix} + \begin{bmatrix}\mathbf{u} \\ u^*\end{bmatrix} R_t\right) \\
&= \begin{bmatrix} \mathbf{w}_t \\ \left(R_t - \mathbf{v}^T\mathbf{w}_t\right)/v^* \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{w}_t \\ \left(R_t - (\tilde{V}(s_t) - \gamma\tilde{V}(s_{t+1}))\right)/v^* \end{bmatrix}
\end{aligned} \tag{8}$$

RITKÍTÁS

RITKÍTÁS

A figyelembe vett adat-pontok száma korlátozható különböző ritkítás segítségével.

RITKÍTÁS

A figyelembe vett adat-pontok száma korlátozható különböző ritkítás segítségével.

Következmények:

- Lesznek olyan esetek amikor a $\tilde{A}$ és $\tilde{\mathbf{b}}$ mérete nem változik.

RITKÍTÁS

A figyelembe vett adat-pontok száma korlátozható különböző ritkítás segítségével.

Következmények:

- Lesznek olyan esetek amikor a $\tilde{A}$ és $\tilde{\mathbf{b}}$ mérete nem változik.
- A frissítés egyenletei változnak

RITKÍTÁS

A figyelembe vett adat-pontok száma korlátozható különböző ritkítás segítségével.

Következmények:

- Lesznek olyan esetek amikor a $\tilde{A}$ és $\tilde{\mathbf{b}}$ mérete nem változik.
- A frissítés egyenletei változnak

RITKÍTÁS

A figyelembe vett adat-pontok száma korlátozható különböző ritkítás segítségével.

Következmények:

- Lesznek olyan esetek amikor a $\tilde{A}$ és $\tilde{\mathbf{b}}$ mérete nem változik.
- A frissítés egyenletei változnak

$$
\begin{aligned}
C_{t+1} =& A_{t+1}^{-1} = \left(A_t + \mathbf{u}\mathbf{v}^T\right)^{-1} \\
=& A_t^{-1} - \frac{1}{Z}\left(A_t^{-1}\mathbf{u}\mathbf{v}^T A_t^{-1}\right) \quad Z = 1 + \mathbf{v}^T A_t^{-1}\mathbf{u} \\
=& C_t - \frac{C_t\mathbf{u}\mathbf{v}^T C_t}{1 + \mathbf{v}^T C_t \mathbf{u}}
\end{aligned}
$$

A közelítési együtthatók frissítése

$$
\begin{aligned}
w_{t+1} &= \left(C_t - \frac{C_t \mathbf{u} \mathbf{v}^T C_t}{1 + \mathbf{v}^T C_t \mathbf{u}} \right) (\mathbf{b}_t + \mathbf{u}\, R_t) \\
&= w_t + C_t \mathbf{u} R_t - \frac{C_t \mathbf{u} \mathbf{v}^T \left(w_t + C_t \mathbf{u} R_t \right)}{1 + \mathbf{v}^T C_t \mathbf{u}}
\end{aligned} \tag{9}
$$

Nincs szükségünk a $\tilde{\mathbf{b}}$ vektor tárolására, hiszen az együtthatók kifejezhetőek nélküle is.

KÖVETKEZMÉNYEK

A módszer előnyei

- Számítási költségek csökkenése: az n dimenziós mátrix invertálásának elkerülése miatt

A módszer előnyei

- Számítási költségek csökkenése: az n dimenziós mátrix invertálásának elkerülése miatt
- Nincs szükségünk az A mátrixot tárolni a memóriában (*elegendő az inverze*)

A módszer előnyei

- Számítási költségek csökkenése: az n dimenziós mátrix invertálásának elkerülése miatt
- Nincs szükségünk az A mátrixot tárolni a memóriában (*elegendő az inverze*)
- Az iteratív egyenleteknek köszönhetően a módszer teljes mértékben használható on-line

A módszer előnyei

- Számítási költségek csökkenése: az n dimenziós mátrix invertálásának elkerülése miatt
- Nincs szükségünk az A mátrixot tárolni a memóriában (*elegendő az inverze*)
- Az iteratív egyenleteknek köszönhetően a módszer teljes mértékben használható on-line
- Lehetőség nyílik a tanuló adatok újrahasznosítására

A módszer előnyei

- Számítási költségek csökkenése: az n dimenziós mátrix invertálásának elkerülése miatt
- Nincs szükségünk az A mátrixot tárolni a memóriában (*elegendő az inverze*)
- Az iteratív egyenleteknek köszönhetően a módszer teljes mértékben használható on-line
- Lehetőség nyílik a tanuló adatok újrahasznosítására

KÖVETKEZMÉNYEK

A módszer előnyei

- Számítási költségek csökkenése: az n dimenziós mátrix invertálásának elkerülése miatt
- Nincs szükségünk az A mátrixot tárolni a memóriában (*elegendő az inverze*)
- Az iteratív egyenleteknek köszönhetően a módszer teljes mértékben használható on-line
- Lehetőség nyílik a tanuló adatok újrahasznosítására

Kérdések

- Hogyan kerülhető el az instabilitás (u^* *és* v^* *tarthat nullához*)
- A kernel hiperparaméterek optimalizálása automatizálható?