

08MIAR-Aprendizaje por refuerzo

Sesión 3 – Algoritmos base: Deep Q-Network



Universidad
Internacional
de Valencia

De:



Planeta Formación y Universidades

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

Proceso de aprendizaje

Algoritmo DQN

Conclusiones

Bibliografía recomendada

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

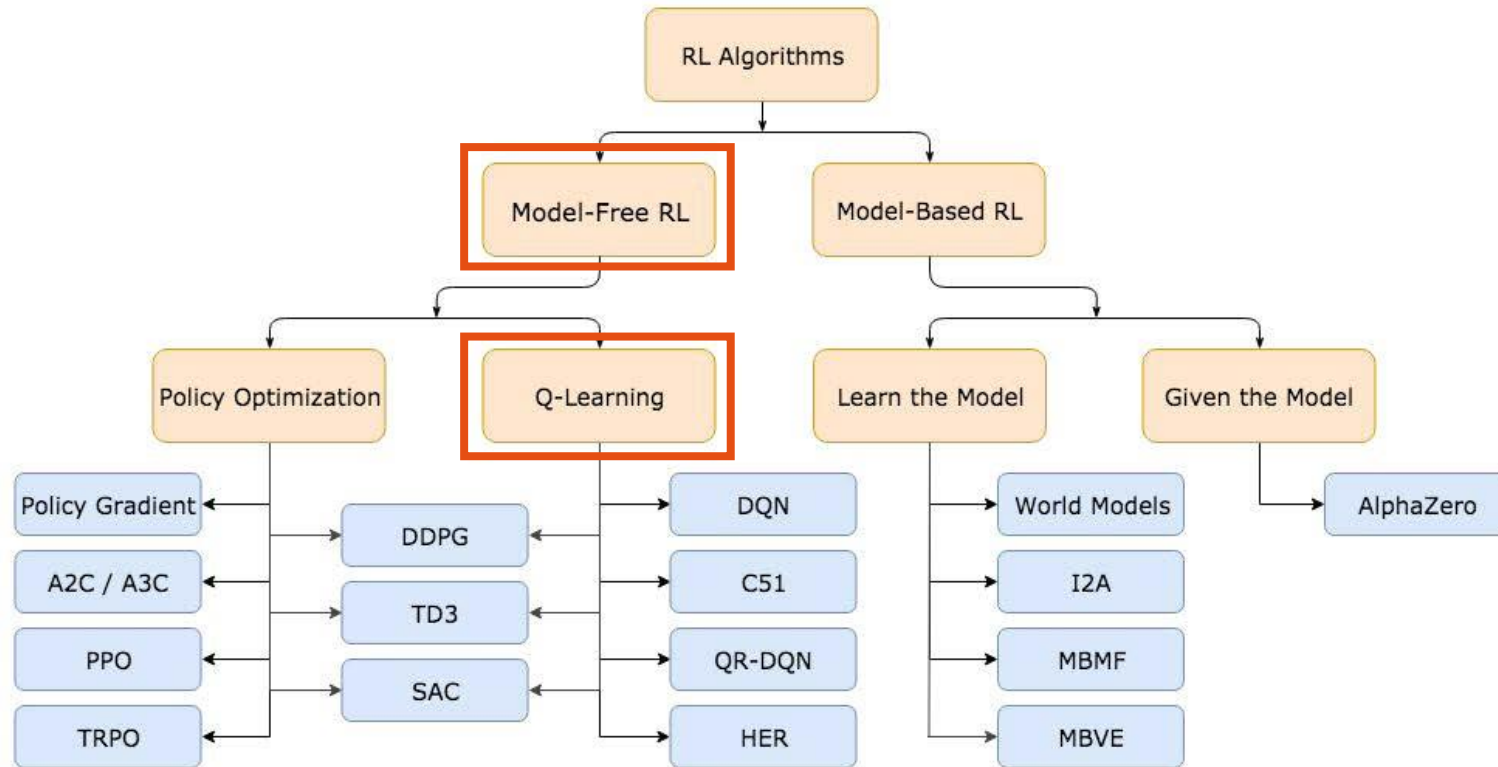
Proceso de aprendizaje

Algoritmo DQN

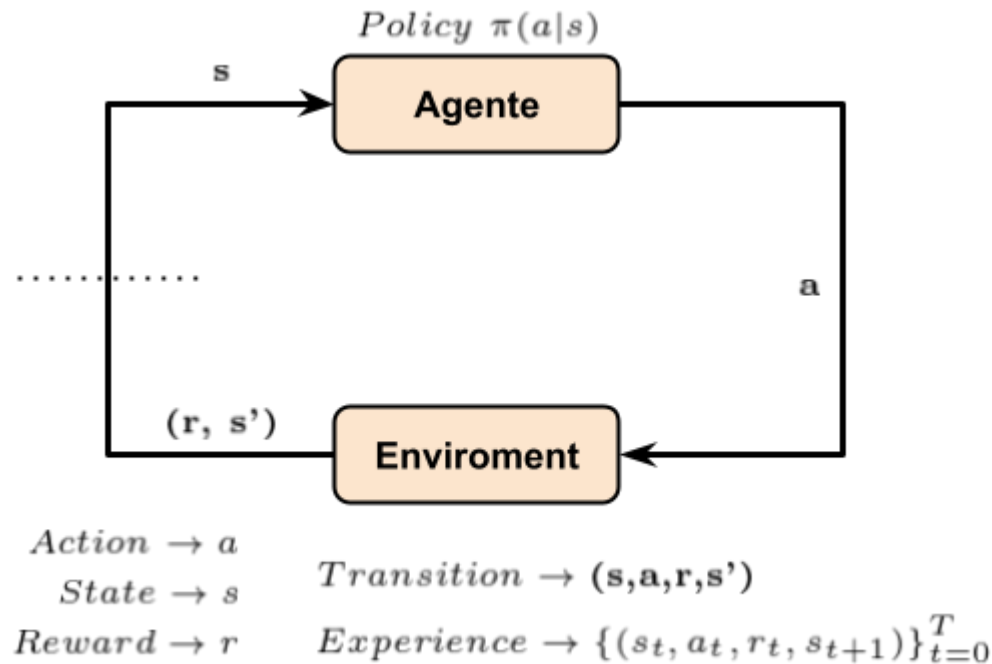
Conclusiones

Bibliografía recomendada

¿Dónde estamos?

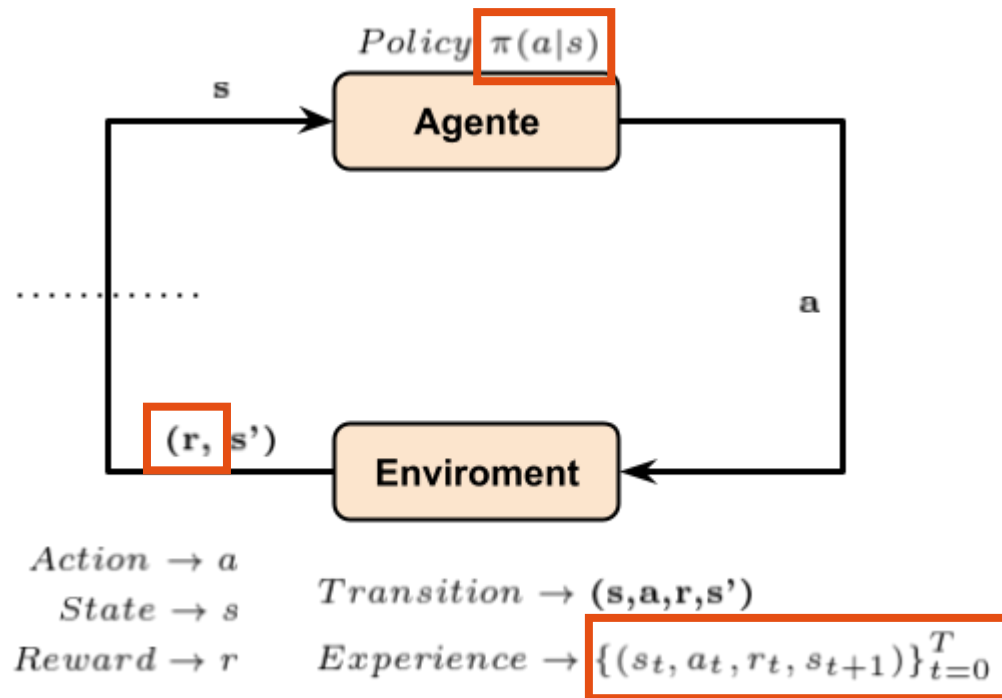


Finite Markov Decision Process



- Finite MDPs: $(\mathcal{S}, \mathcal{A})$ son espacios finitos.
- **Markov property**: La probabilidad de cada valor posible para s' depende del estado y la acción inmediatamente anteriores.
- El **estado** debe incluir información sobre todos los **aspectos relevantes de la interacción agente-entorno en el pasado**.

¿Cómo de buena es una policy?



- Una **policy**, $\pi(a|s)$, es una función de **mapeo de estados a probabilidades sobre acciones**.
- El **objetivo** del agente es **maximizar la recompensa acumulada a futuro**.

$$G_t = r_{t+1} + r_{t+2} + r_{t+3} + \dots + r_T$$

- **Discounted reward**: discount factor $0 \leq \gamma \leq 1$.

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- Propiedad de **recursividad**: $G_t = r_{t+1} + \gamma G_{t+1}$

Valor y calidad de un estado

- **Función value, $v_{\pi}(s)$.** La función estado-valor de un estado, s , **bajo una policy π** , es la **recompensa esperada a futuro** comenzando en s , y siguiendo la policy desde entonces.

$$v_{\pi}(s) = E_{\pi}[G_t|s] = \sum_a \pi(a|s) \cdot r_{t+1} + v_{\pi}(s')$$

- **Quality, $q_{\pi}(s, a)$.** La función calidad de tomar la acción a , en estado s , bajo una , **bajo una policy π** , **es la recompensa esperada a futuro**, comenzando en s , tomando la acción a , y a continuación siguiendo la policy.

$$q_{\pi}(s, a) = E_{\pi}[G_t|(s, a)] = \{ r_{t+1} + v_{\pi}(s') \}_a$$

Q-learning

- **Política óptima, π^* .** Aquella estrategia que, siguiéndola, nos devuelve la **máxima recompensa esperada a futuro**. Se suele seguir una **política voraz (*greedy*)**, respecto a $q_\pi(s, a)$.

$$\pi^*(s) = \operatorname{argmax}_a (q^{\pi^*}(s, a))$$

- **Ecuación de Bellman.** El objetivo entonces es **encontrar la función q óptima** para tu problema, y luego seguir una política voraz sobre dicha función. La ecuación de Bellman proporciona una **definición recursiva** con la que podremos encontrar la función óptima q .

$$q^{\pi^*}(s, a) = r + \gamma \max_{a'} q^{\pi^*}(s', a')$$

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

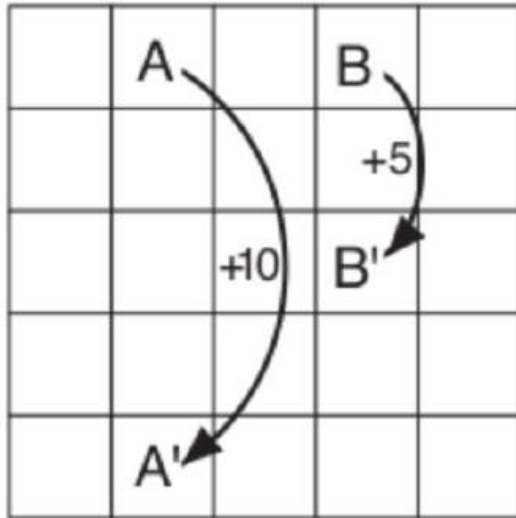
Proceso de aprendizaje

Algoritmo DQN

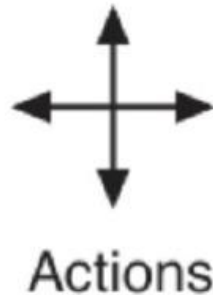
Conclusiones

Bibliografía recomendada

Gridworld



Gridworld de Sutton & Barto, Capítulo 3



Gridworld de Sutton & Barto, Capítulo 3

- Las **celdas** corresponden con los **estados** del entorno.
- La recompensa para cada acción-estado es:
 - **+10** para la transición de **A a A'** (que es la única acción que se puede ejecutar en A).
 - **+5** para la transición de **B a B'** (que es la única acción que se puede ejecutar en B).
 - **-1** si el movimiento se sale del *grid*.
 - **0** en cualquier otro caso.

Random policy evaluation

- Originalmente, la estimación de q se realiza por medio de **programación dinámica**, de forma **iterativa**. De esta forma, se define una **política aleatoria**, y se realiza una exploración del entorno actualizando la función v en base a la experiencia.
- Política aleatoria, $\pi^r(s|a) = 1/A$.

i) Inicialización $v_\pi(s)$

```
array([ 0., 0., 0., 0., 0.],
      [0., 0., 0., 0., 0.],
      [0., 0., 0., 0., 0.],
      [0., 0., 0., 0., 0.],
      [0., 0., 0., 0., 0.]])
```

ii) Exploración iterativa

- Comenzamos en $s=[0,0]$. Para cada acción:

- Arriba: $0.25 * (-1 + 0.9 * 0) = -0.25$
- Abajo: $0.25 * (0 + 0.9 * 0) = 0$
- Derecha: $0.25 * (0 + 0.9 * 0) = 0$
- Izquierda: $0.25 * (-1 + 0.9 * 0) = -0.25$

$$v_\pi(s) = E_\pi[G_t|s] = \sum_a \pi(a|s) \cdot r_{t+1} + v_\pi(s') = -0.5$$

- Repetir para cada **iteración k** , para **todo s** , y actualizar la función value tras cada iteración $v_\pi(s)$

Exploración iterativa

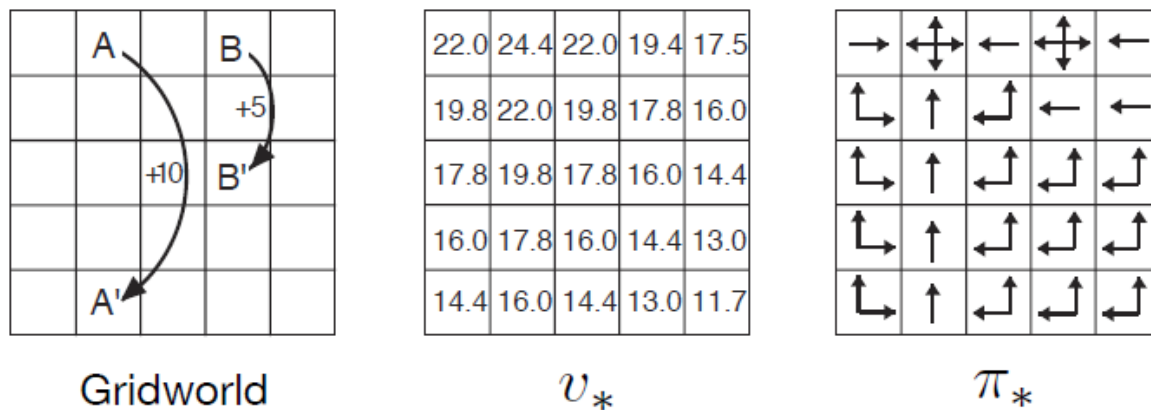
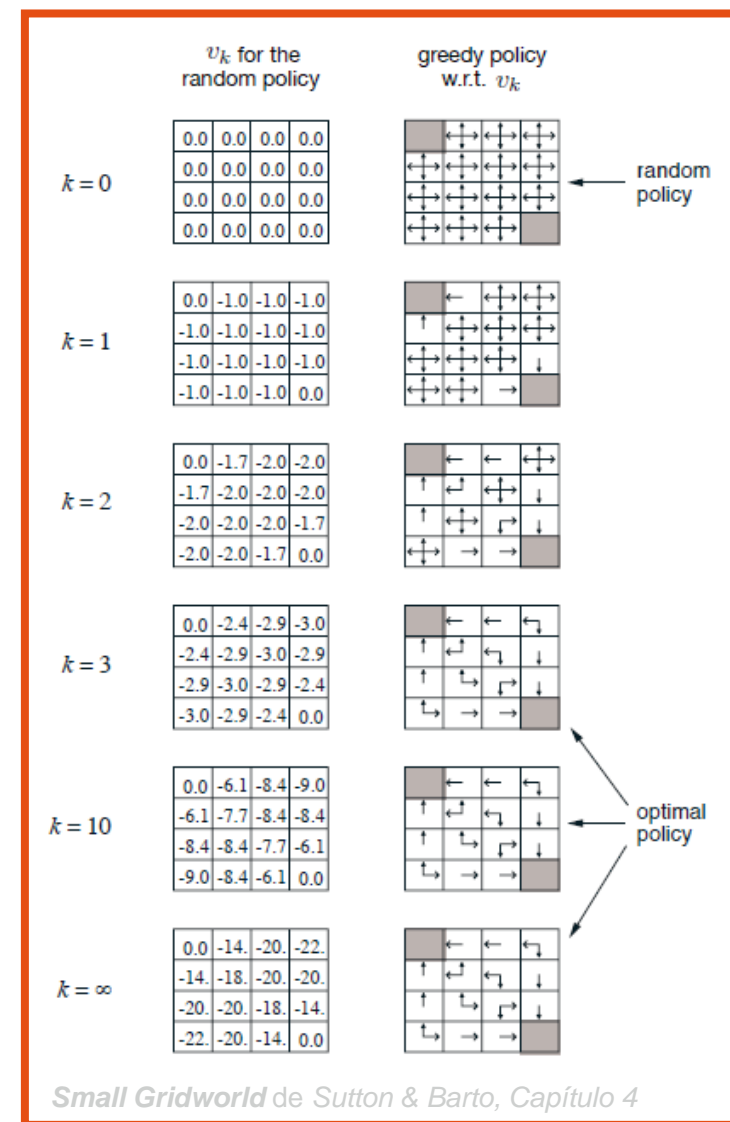


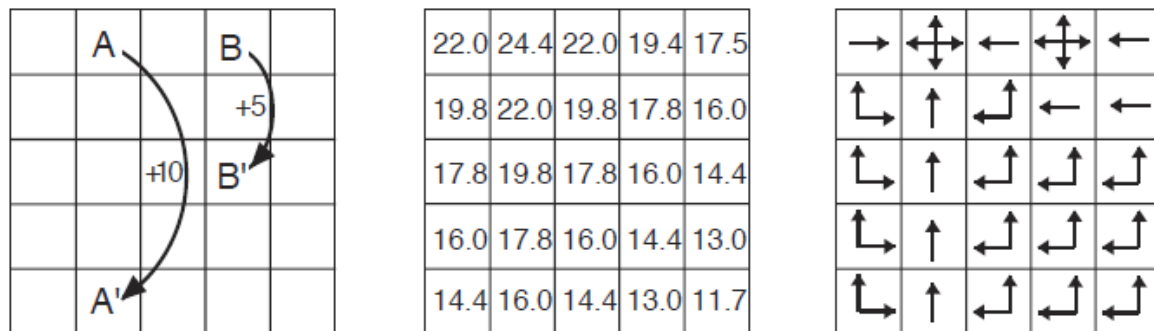
Figure 3.5: Optimal solutions to the gridworld example.

Gridworld de Sutton & Barto, Capítulo 3



Small Gridworld de Sutton & Barto, Capítulo 4

Exploración iterativa



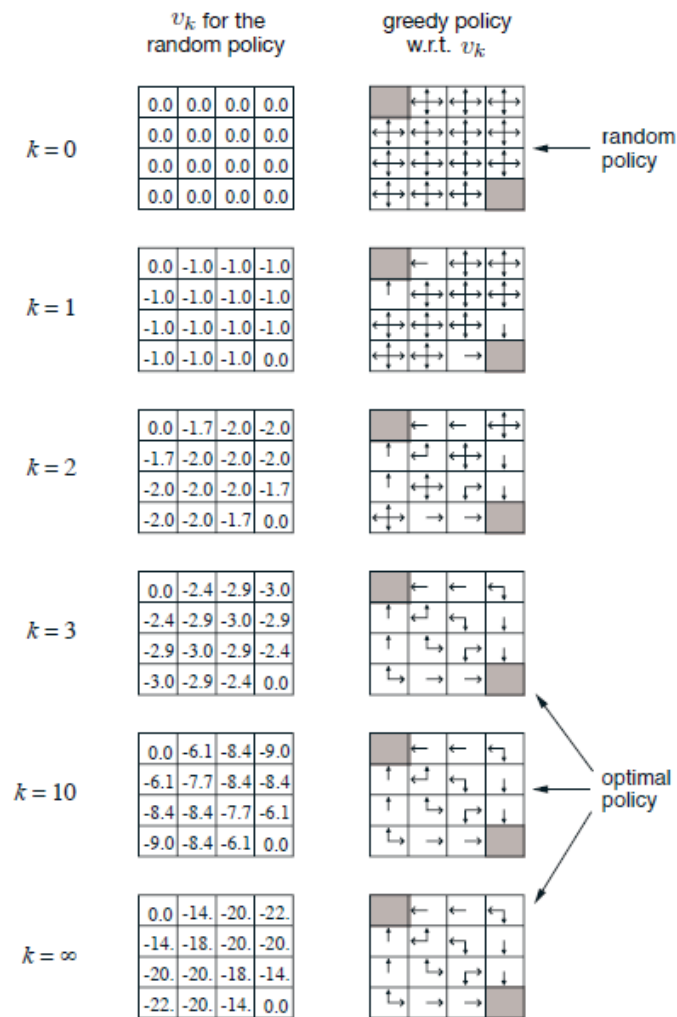
Gridworld

v_*

π_*

Figure 3.5: Optimal solutions to the gridworld example.

Gridworld de Sutton & Barto, Capítulo 3



Small Gridworld de Sutton & Barto, Capítulo 4

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

Proceso de aprendizaje

Algoritmo DQN

Conclusiones

Bibliografía recomendada

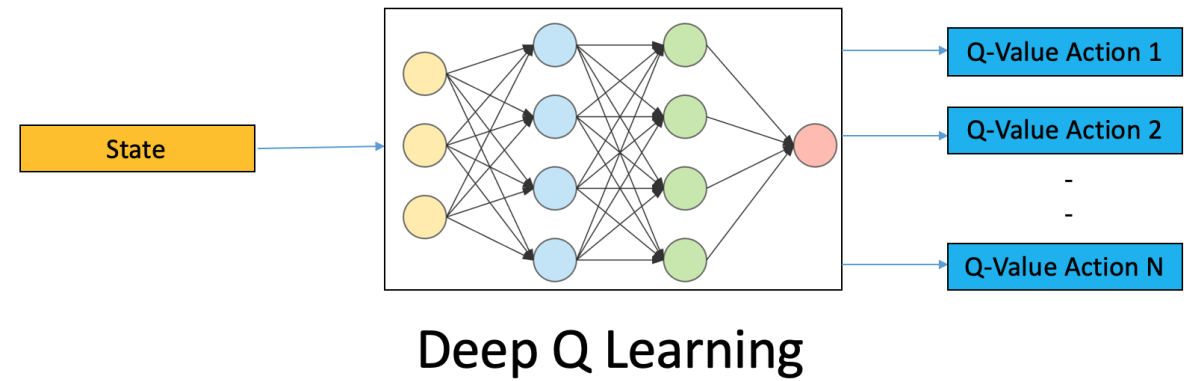
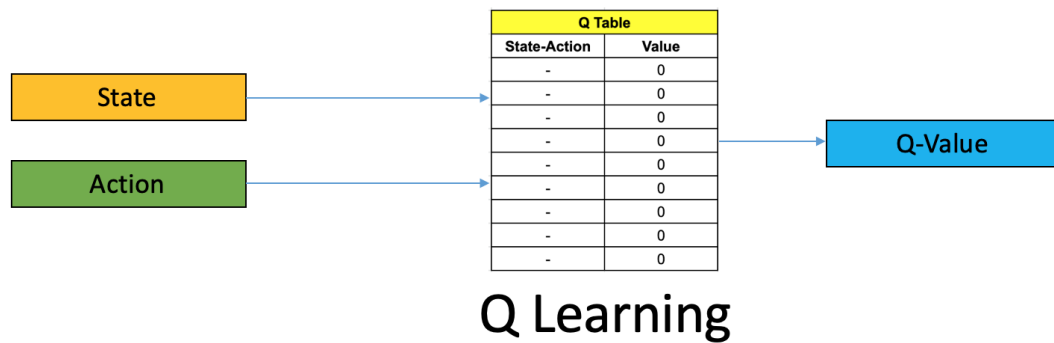
Deep Q-network

- Asunciones en *Finite MDPs* : $(\mathcal{S}, \mathcal{A})$ son espacios finitos.
- En escenarios prácticos, la **dimensionalidad** del espacio de **estados y acciones** hace **intratable** esta evaluación iterativa.
- Uso de una **DNN**, $f_{\theta}(\cdot)$, como función **aproximadora** de $q^{\pi^*}(s, a)$.

$$\pi^*(s) = \operatorname{argmax}_a \left(q^{\pi^*}(s, a) \right) = \operatorname{argmax}_a \left(f_{\theta}(s) \right)$$

- Debido a que la mayoría de las simulaciones trabajan con la pantalla directamente, el tipo de red neuronal que usaremos serán **redes convolucionales**.

Deep Q-network



<https://cdn.analyticsvidhya.com/wp-content/uploads/2019/04/Screenshot-2019-04-16-at-5.46.01-PM.png>

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

Proceso de aprendizaje

Algoritmo DQN

Conclusiones

Bibliografía recomendada

Proceso de aprendizaje

- Uso de **Bellman** como función aproximadora en nuestro proceso de aprendizaje.

$$q(s, a) = r + \gamma \max_a q(s', a)$$

- Considerando que $q(s, a)$ como una red neuronal, $f_{\theta}(\cdot)$, se puede definir una **función de coste** en base a las **diferencias temporales**, y optimizar la red neuronal en base a la tupla (s, a, r, s') de cada iteración por **descenso de gradiente**. Como función de pérdidas, el error cuadrático suele ser la solución base.

$$q(s, a) \rightarrow r + \gamma \max_a q(s', a)$$

$$L = (r + \gamma \max_a q(s', a) - q(s, a))^2$$

Proceso de aprendizaje

- El uso de una **red neuronal para estimar la función $q(s, a)$** no asegura convergencia a la $q^{\pi^*}(s, a)$. Es un **proceso inestable**, con tendencia a llegar a **mínimos locales**. Se han desarrollado una serie de **estrategias** para aliviar esto:



- Series temporales.**
- Exploración.** Se sigue una política ϵ -greedy para evitar sobreajuste.
- Experience replay.** En la experiencia se almacenan tuplas (s, a, r, s') obtenidas con versiones antiguas de la red $q(s, a)$.
- Target network.** En la función de coste se sustituye la $q(s, a)$ del siguiente estado por una versión anterior de la red neuronal, llamada target network, $\hat{q}(s', a)$. That is: $q(s, a) \rightarrow r + \gamma \max_a \hat{q}(s', a)$.
- Error clipping.** Evitar actualizaciones demasiado altas. Se suele utilizar la función de Huber.

$$Huber(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \leq 1, \\ (|a| - \frac{1}{2}), & \text{otherwise.} \end{cases}$$

Por qué DQN es off-policy?

- **DQN** es el principal exponente de la familia de métodos conocidos como **off-policy**. Pero, ¿a qué se refiere este tipo de aprendizaje? Os encontraréis dos interpretaciones en la literatura:
 - Basada en **Exploración**. Al seguir una **política ϵ -greedy**, la política que tratamos de predecir como **política objetivo (voraz, greedy)** no se corresponde con la que nuestro agente está siguiendo.
 - Basada en **Experience replay**. Al acumular en la **experiencia** tuplas (s, a, r, s') de **antiguas iteraciones**, la red es actualizada con políticas que no se corresponden con la actual.
- La opción más aceptada en la literatura es la segunda, basada en la **clasificación de estrategias de aprendizaje centrada en el uso de la memoria**.

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

Proceso de aprendizaje

Algoritmo DQN

Conclusiones

Bibliografía recomendada

Algoritmo DQN

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory \mathcal{D} to capacity N
 Initialize action-value function Q with random weights
for episode = 1, M **do**
 Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
 for $t = 1, T$ **do**
 With probability ϵ select a random action a_t
 otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
 Execute action a_t in emulator and observe reward r_t and image x_{t+1}
 Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
 Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in \mathcal{D}
 Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from \mathcal{D}
 Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
 Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
 end for
end for

Playing Atari with Deep Reinforcement Learning (<https://arxiv.org/abs/1312.5602>)

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

Proceso de aprendizaje

Algoritmo DQN

Conclusiones

Bibliografía recomendada

Conclusiones

- Hemos estudiado el primero de los algoritmos base que veremos en la asignatura, **DQN**. Este algoritmo pertenece a la familia de **métodos *off-policy***.
- Está basado en **Q-learning**, donde el objetivo es **estimar la recompensa** esperada para cada par **estado/acción**.
- **DQN combina Q-learning con arquitecturas de Deep Learning**, para poder abarcar problemas con **espacios de dimensiones muy grandes**
- Algunas variaciones, necesarias para la convergencia de la solución, en la versión final del algoritmo de DQN son el uso de una **target network** y de **secuencias de frames** como datos de entrada del modelo del agente.

Índice

Definición Q-learning

Ejemplo Q-learning: gridworld

Deep Q-network

Proceso de aprendizaje

Algoritmo DQN

Conclusiones

Bibliografía recomendada

Bibliografía recomendada

- Human level control through Deep Reinforcement learning, Google Deepmind
<https://deepmind.com/research/publications/2019/human-level-control-through-deep-reinforcement-learning>
- An Introduction to Q-learning: Reinforcement Learning, Sayak Paul, Floydhub
<https://blog.floydhub.com/an-introduction-to-q-learning-reinforcement-learning/>



viu

Universidad
Internacional
de Valencia

universidadviu.com

De:
 Planeta Formación y Universidades