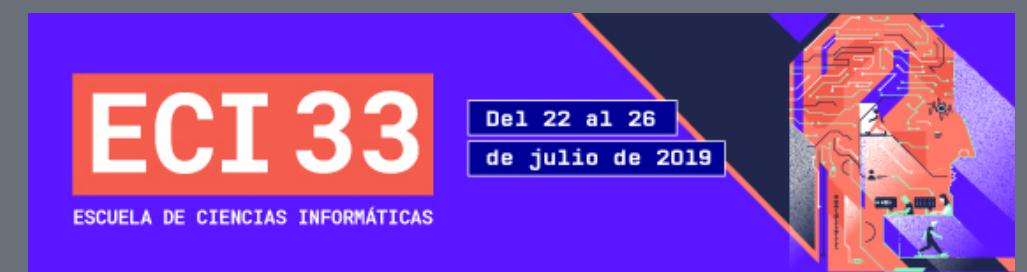


Aprendizaje Profundo por Refuerzo

04. Aprendizaje profundo por refuerzo

Dr. Juan Gómez Romero
Investigador Senior
Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad de Granada



Aprendizaje profundo por refuerzo

Bibliografía



UNIVERSIDAD
DE GRANADA

R.S. Sutton, A.G. Barto (2018) Reinforcement Learning. MIT Press.

<https://mitpress.mit.edu/books/reinforcement-learning-second-edition>

A. Zai, B. Brown (2018) Deep Reinforcement Learning in Action. Manning.

M. Morales (2018) Grokking Deep Reinforcement Learning. Manning.

M. Pumperla, K. Ferguson (2019). Deep Learning and the Game of Go. Manning.

L. Weng (2018) A (Long) Peek into Reinforcement Learning. <https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html>

OpenAI (2019) Spinning Up in Deep RL. <https://spinningup.openai.com/en/latest/>

Índice

1. Deep Q-Learning
2. Métodos basados en política
3. Métodos *actor-critic*
4. AlphaGo
5. Aplicaciones

1 Deep Q-Learning

FORMALIZACIÓN, IMPLEMENTACIÓN

Deep Q-Learning

Q-Learning



UNIVERSIDAD
DE GRANADA

Estimación $Q(s, a)$

Estado	Acción		
	A_1	\dots	A^m
S^1			
\dots			
S^n			

Problemas con espacio de estados o acciones continuas

No se pueden representar en una tabla Q

A partir de secuencias de episodios

$S_0 \ A_0 \ R_1 \ S_1$

$A_1 \ R_2 \ S_2$

$$\rightarrow Q(S_0, A_0) \leftarrow Q(S_0, A_0) + \alpha \left(R_1 + \gamma \max_{a \in \mathcal{A}} Q(S_1, a) - Q(S_0, A_0) \right)$$
$$\pi \leftarrow \epsilon_0\text{-greedy}(Q)$$

$$Q(S_1, A_1) \leftarrow Q(S_1, A_1) + \alpha \left(R_2 + \gamma \max_{a \in \mathcal{A}} Q(S_2, a) - Q(S_1, A_1) \right)$$
$$\pi \leftarrow \epsilon_0\text{-greedy}(Q)$$

Experiencias se generan de forma secuencial

Dependencia entre *experiencias*
Algunas zonas son difíciles de explorar

Experiencias se descartan

Si aparece una *experiencia* interesante solo se utiliza una vez en el proceso de aprendizaje

Deep Q-Learning

Q-Learning



UNIVERSIDAD
DE GRANADA

Estimación $Q(s, a)$

$$f(s, a) \rightarrow \mathbb{R}$$

Modelar Q como una función no lineal

Se puede trabajar con estados y acciones continuos

A partir de pasos no secuenciales

$S_0 \ A_0 \ R_1 \ S_1$

$S_1 \ A_1 \ R_2 \ S_2$

...

Guardar una base de datos o memoria de *experiencias (rolling)*

Se puede utilizar en cualquier orden

Se puede utilizar más de una vez

Deep Q-Learning

Estimación de Q con red neuronal

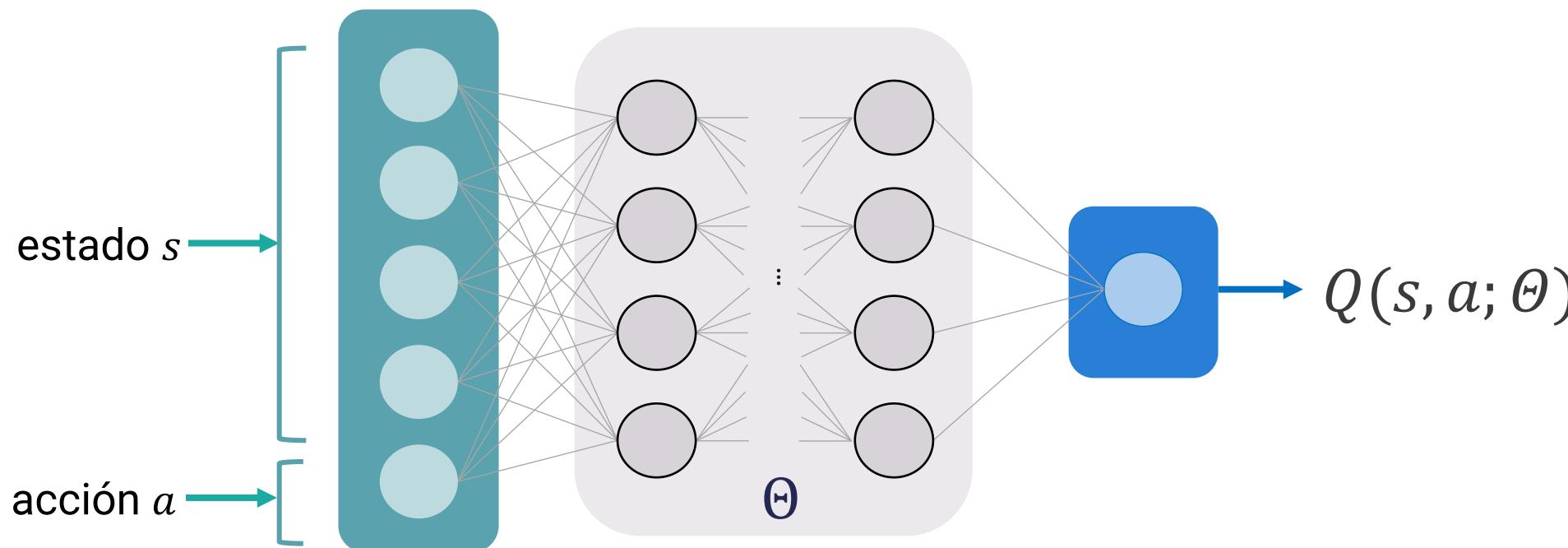


UNIVERSIDAD
DE GRANADA

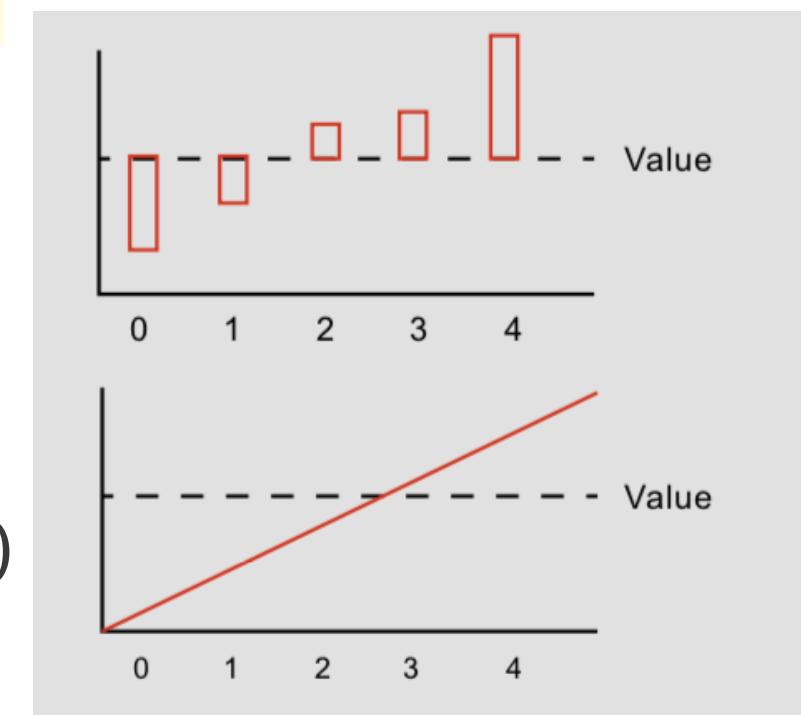
Estimación $Q(s, a)$

$$f(s, a) \rightarrow \mathbb{R}$$

Modelar Q como una función no lineal
Se puede trabajar con estados y acciones continuos



$$Q(s_0, a) = [-2.0, -1.0, 0.5, 1.0, 3.0]$$



$$Q(s_0, a; \Theta)$$

Deep Q-Learning

Estimación de Q con red neuronal



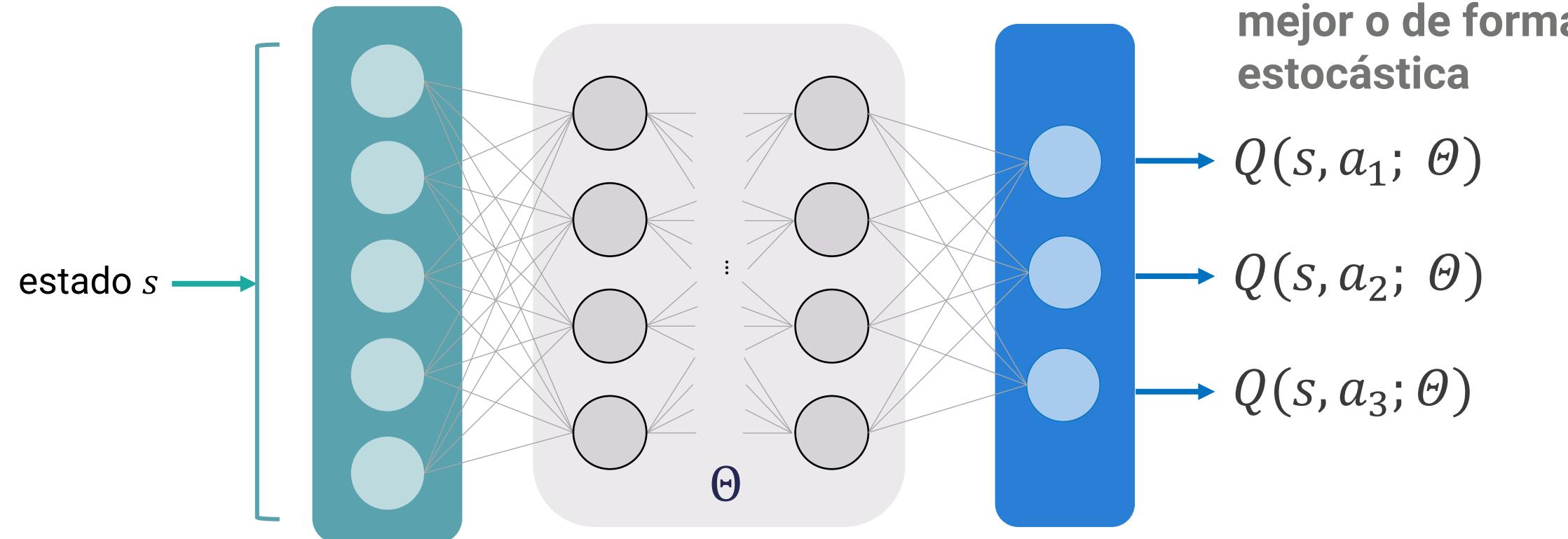
UNIVERSIDAD
DE GRANADA

Estimación $Q(s, a)$

$$f(s, a) \rightarrow \mathbb{R}$$

Modelar Q como una función no lineal

Se puede trabajar con estados y acciones continuas



**Seleccionar la
mejor o de forma
estocástica**

$$Q(s, a_1; \Theta)$$

$$Q(s, a_2; \Theta)$$

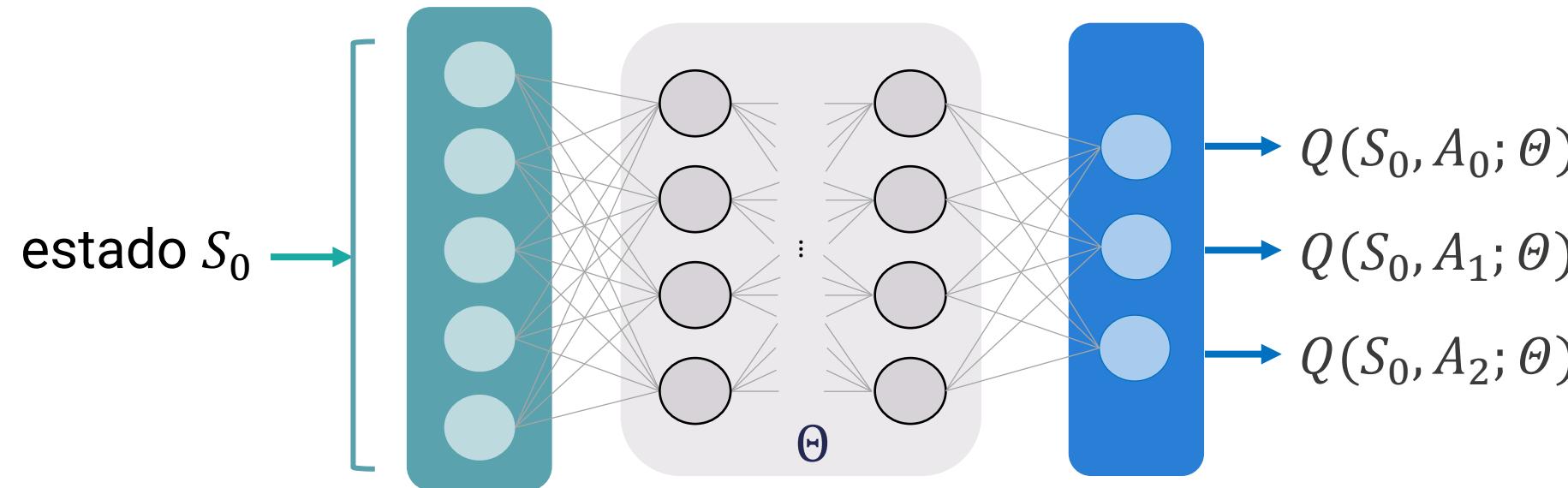
$$Q(s, a_3; \Theta)$$

Deep Q-Learning

DQN – Deep Q-Network



UNIVERSIDAD
DE GRANADA



Entrenamiento (*learning*)

Entrada: S

Salida: $Q(S; \theta)$

Función de error: $\mathcal{L}(Q(S), Q(S; \theta))$

*¡No disponemos
de esta salida!*

Guardar una base de datos o memoria de
experiencias

Se puede utilizar en cualquier orden y más de una vez

$$Q(S_0, A_0) \leftarrow Q(S_0, A_0) + \alpha \left(R_1 + \gamma \max_{a \in \mathcal{A}} Q(S_1, a) - Q(S_0, A_0) \right)$$

*Pero podemos estimarla a partir de:
 $S_0 \ A_0 \ R_1 \ S_1$*

Deep Q-Learning

Formulación DQN

$$\ell(x^{(i)}, \Theta) = \mathcal{L}(Q(S), Q(S, A; \Theta))$$

Con MSE: $E = \mathbb{E}_{\pi} [(Q(S, A) - Q(S, A; \Theta))^2]$

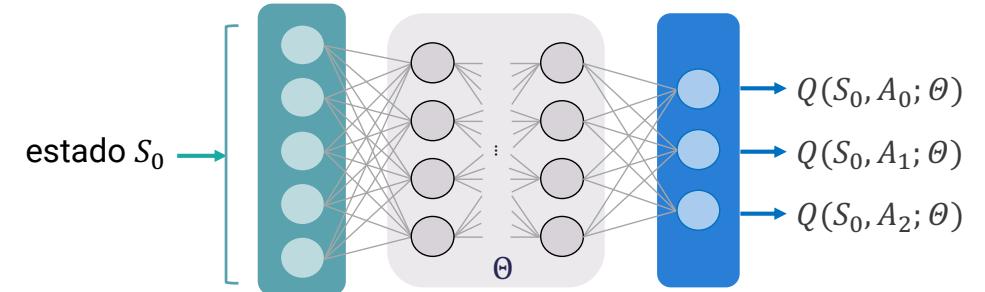
Recordemos que: $\Delta\Theta = \Delta w = -\eta \frac{\partial E}{\partial w}$

Desarrollamos: $\frac{\partial E}{\partial w} = 2(Q(S, A) - Q(S, A; \Theta)) \frac{\partial Q(S, A; \Theta)}{\partial w}$ gradiente de los pesos red ∇_{Θ}

Sustituimos: $\Delta\Theta = -2\eta (Q(S, A) - Q(S, A; \Theta)) \nabla_{\Theta} Q(S, A; \Theta)$

Simplificamos y aplicamos la expresión de Q-Learning: $Q(S_0, A_0) \leftarrow Q(S_0, A_0) + \alpha \left(R_1 + \gamma \max_{a \in \mathcal{A}} Q(S_1, a) - Q(S_0, A_0) \right)$

$$\Delta\Theta = \alpha \left(R + \gamma \max_{a \in \mathcal{A}} Q(S', a; \Theta) - Q(S, A; \Theta) \right) \nabla_{\Theta} Q(S, A; \Theta)$$



Deep Q-Learning

Formulación DQN



UNIVERSIDAD
DE GRANADA

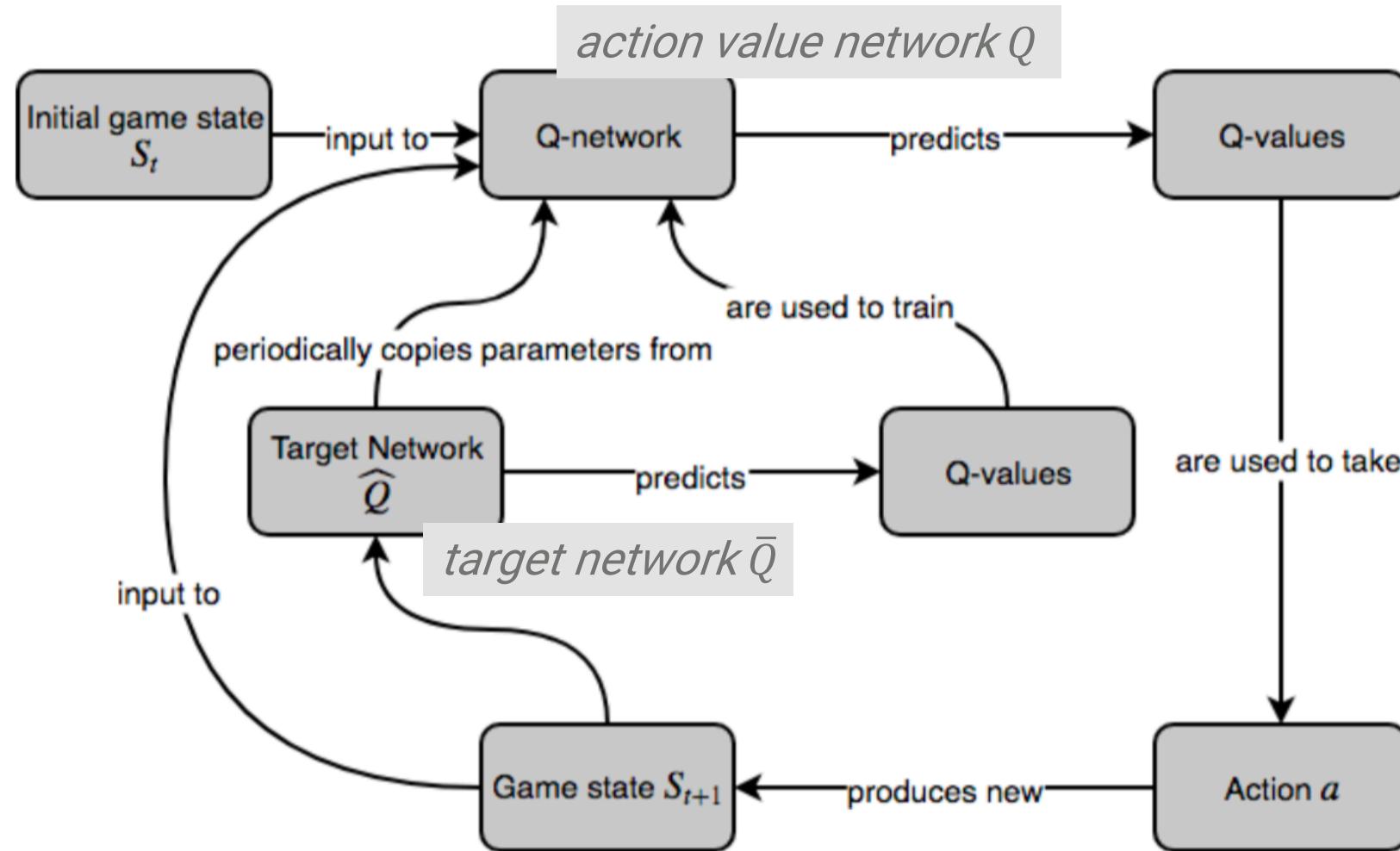
¿Qué significa esto?

- La expresión para **actualizar los pesos de la red es diferente** al algoritmo de gradiente descendente que vimos en el Tema 01
- Para actualizar los pesos de la red Q usando gradientes **es posible utilizar un *mini-batch*** extraido de la memoria de experiencias; esto es, un sub-conjunto $S_t \ A_t \ R_{t+1} \ S_{t+1}$
- Estamos actualizando una aproximación con una aproximación. Por ello, **se suelen utilizar dos redes Q** (misma topología; los pesos Θ^- de la red *target* \bar{Q} se actualizan con menor frecuencia a partir de la red *action value* Q)

$$\Delta\Theta = \alpha \left(R + \gamma \max_{a \in \mathcal{A}} \bar{Q}(S', a; \Theta^-) - Q(S, A; \Theta) \right) \nabla_\Theta Q(S, A; \Theta)$$

Deep Q-Learning

Algoritmo DQN



A. Zai, B. Brown (2018) *Deep Reinforcement Learning in Action*. Manning.



V. Mnih et al. (2015) *Human-level control through deep reinforcement learning*. Nature 518, 529-533

Deep Q-Learning

Algoritmo DQN



UNIVERSIDAD
DE GRANADA



Deep Mind DQN (2016). Más: <https://deepmind.com/research/dqn/>

Deep Q-Learning

Algoritmo DQN



UNIVERSIDAD
DE GRANADA

Deep Q-Learning

params: Q y \bar{Q} iniciales, D inicial, $n_{\text{episodios}}$, $n_{\text{entrenamiento}}$, n_{batch} , γ , C , τ

for $i = \{0, \dots, n_{\text{episodios}}\}$

 for $t = \{0, \dots, n_{\text{entrenamiento}}\}$

 elegir acción A_t con política ϵ -greedy(Q)

sample

 aplicar A_t y obtener R_{t+1}, S_{t+1}

 almacenar $S_t A_t R_{t+1} S_{t+1}$ en memoria D

 muestrear un mini-batch de experiencias $\langle S_j A_j R_{j+1} S_{j+1} \rangle$ de tamaño n_{batch} de la memoria D

 obtener salida de predicción de cada experiencia:

$$y_j = \begin{cases} R_j & \text{(si el episodio termina en } j+1, \text{ o en otro caso)} \\ R_j + \gamma \max_{a \in \mathcal{A}} \bar{Q}(S_{j+1}, a; \theta^-) & \end{cases}$$

 optimizar mediante gradiente descendente sobre $(y_j - Q(S_j, A_j; \theta))^2$

update \bar{Q}

 cada C iteraciones, actualizar \bar{Q} a partir de Q ; por ejemplo, $\theta^- = \tau \theta^- + (1 - \tau) \theta$

Deep Q-Learning

Avanzado



UNIVERSIDAD
DE GRANADA

Convergencia



- Z. Yang, Y. Xie, Z. Wang (2019) *A Theoretical Analysis of Deep Q-Learning*. <https://arxiv.org/abs/1901.00137v2>
S.J. Lee (2019) *The Deep Q-Network Book*. <https://www.dqnbook.com>

Double DQN

DQN tiende a sobreestimar los valores de Q , sobre todo al principio del algoritmo >> Evaluar cómo de buena es A_t con θ^-

Memoria de experiencias con prioridad

La memoria tiene un tamaño limitado >> Mantener experiencias antiguas valiosas
La memoria es muestrada de forma uniforme >> Definir probabilidad no uniforme

Dueling networks

Utilizar una arquitectura diferente, de forma que: $V(s) = Adv(s) + Q(s, a)$, donde $Adv(s)$ es la *función de ventaja*

Otras técnicas

Distributional DQN, Noisy DQN, Rainbow

2 Métodos basados en política

FORMALIZACIÓN

Métodos basados en política

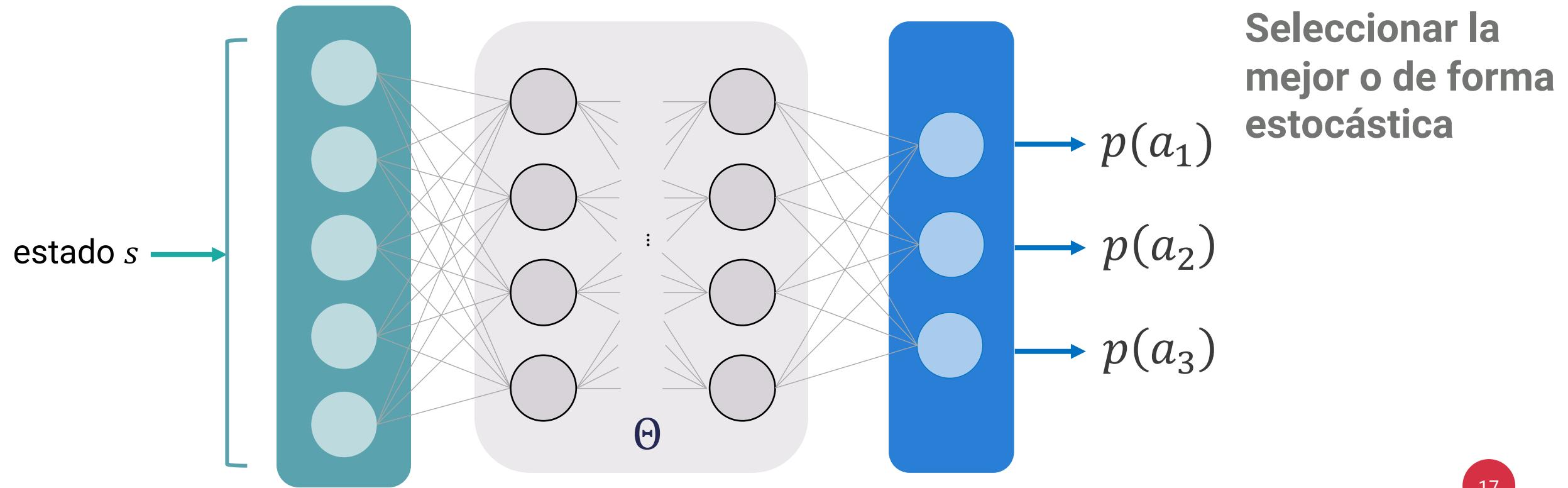
Concepto



UNIVERSIDAD
DE GRANADA

Hasta ahora, hemos intentado estimar Q , y a partir de la ésta, conseguir la política óptima

¿Por qué no intentar calcular directamente qué acción es mejor, sin necesidad de estimar Q ? **Métodos basados en optimización de política** (vs Métodos basados en estimación de valor)



Métodos basados en política

Hill-climbing

Hill-climbing

params: θ inicial, S_0

$G_{max} \leftarrow 0$, $\theta_{max} \leftarrow \theta$

while no resuelto

 generar episodio según las acciones seleccionadas con θ

 calcular recompensa acumulada G (desde estado inicial)

 actualizar G_{max} , θ_{max}

 modificar aleatoriamente θ



UNIVERSIDAD
DE GRANADA

Mejoras

Enfriamiento simulado

Ruido adaptativo

Programación evolutiva

Métodos basados en política

REINFORCE

REINFORCE

params: θ inicial

while no resuelto

generar K trayectorias $x^{(i)}$ de longitud n según política π_θ

calcular recompensa obtenida R^i para cada $x^{(i)}$

actualizar θ para estimar el gradiente de la función "máximo beneficio" $U(\theta)$

Función $U(\theta)$ a maximizar:

$$U(\theta) = \sum_x P(x; \theta)R(x)$$

Gradiente de $U(\theta)$:

$$\nabla_\theta U(\theta) \approx \frac{1}{K} \sum_{i=1}^K \sum_{t=1}^n \nabla_\theta \log \pi_\theta(A_t^{(i)} | S_t^{(i)}) R(x^{(i)})$$

3 Métodos *actor-critic*

FORMULACIÓN, IMPLEMENTACIÓN

Métodos *actor-critic*

Concepto



UNIVERSIDAD
DE GRANADA

Método basados en política (REINFORCE)

calcular probabilidad de que una acción conduzca a un buen resultado final (ACTUAR)

Problema: una trayectoria que conduzca a un mal resultado final puede contener buenas acciones

+

Métodos basados en valor (DQN)

estimar el valor de recompensa obtenido (ESTIMAR)

Problema: se estima a partir de estimaciones, lo cual introduce sesgo en el aprendizaje

=

Métodos *actor-critic*

Combinan ambas aproximaciones

Utilizan dos redes: *actor* (similar a la red de REINFORCE - π), *critic* (similar a la red de DQN - V)

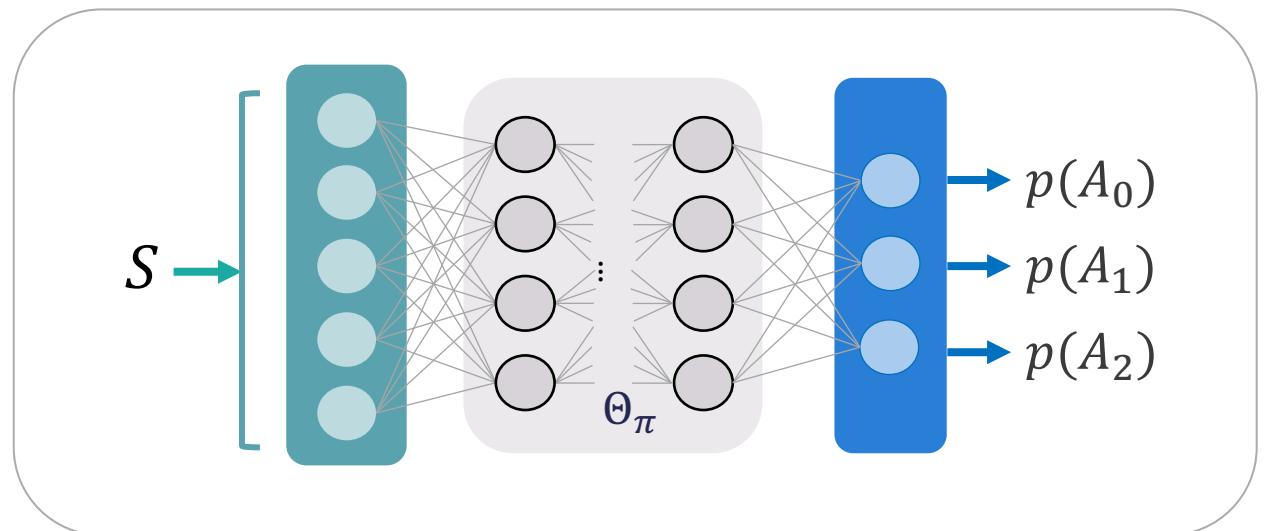
Ventajas: acelerar el aprendizaje (más rápidos), evitar sesgos (más estable)

Métodos *actor-critic*

Concepto

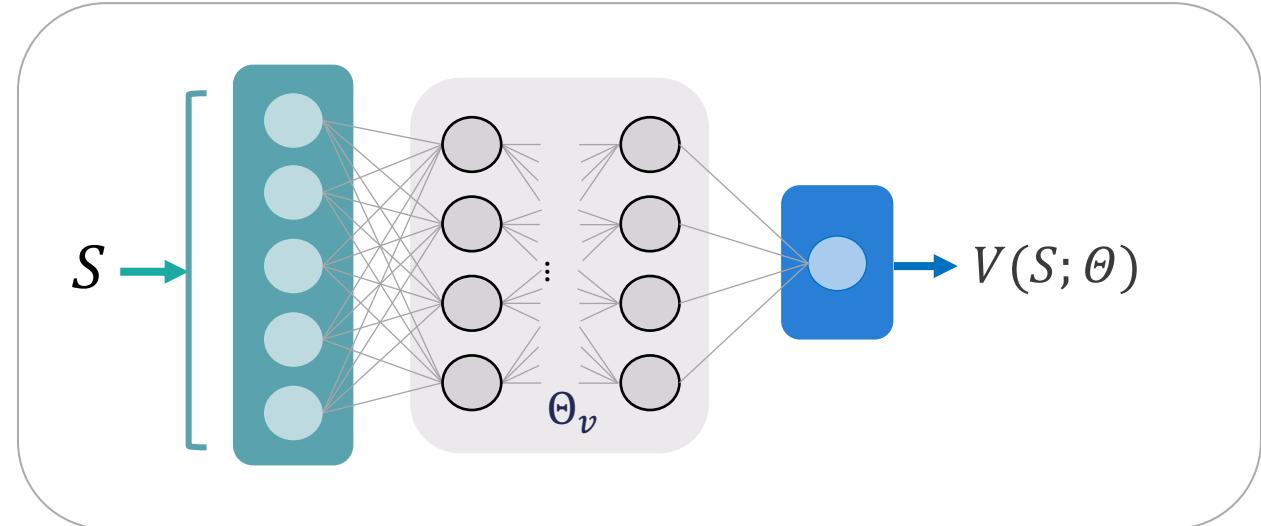
actor

$$\pi(A|S; \theta_\pi)$$



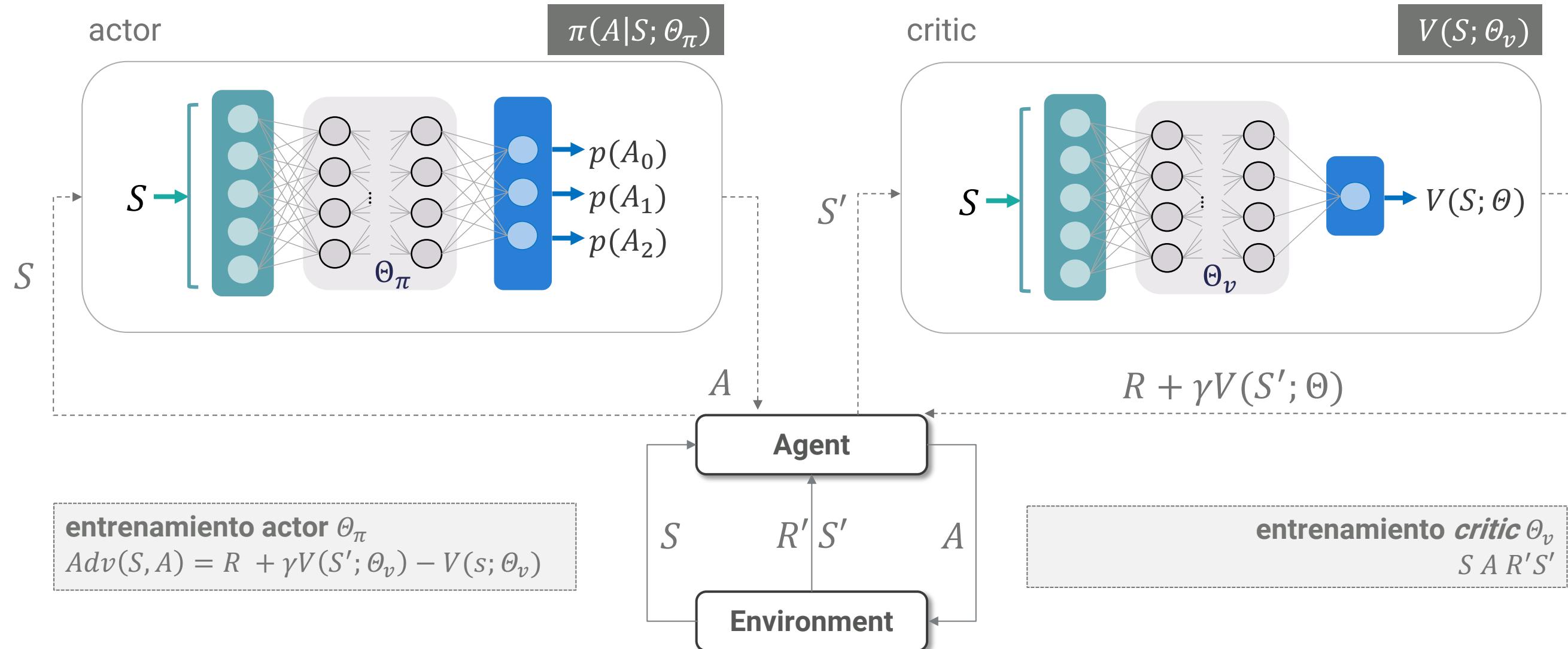
critic

$$V(S; \theta_v)$$



Métodos *actor-critic*

Algoritmo



Mejoras

Arquitectura paralela con múltiples agentes que aprenden simultáneamente, en lugar de implementar memoria de experiencias y mini-batches

- *A3C : Asynchronous Advantage Actor-Critic (2016)*

Cada agente utiliza una copia local de la red *actor-critic*

Periódicamente, cada agente actualiza de forma asíncrona la red *actor-critic* global

- *A2C : Advantage Actor-Critic (2016)*

Los agentes actualizan de forma síncrona la red *actor-critic*



V. Mnih et al. (2016) *Asynchronous Methods for Deep Reinforcement Learning*. <https://arxiv.org/abs/1602.01783>

Métodos *actor-critic*

Avanzado

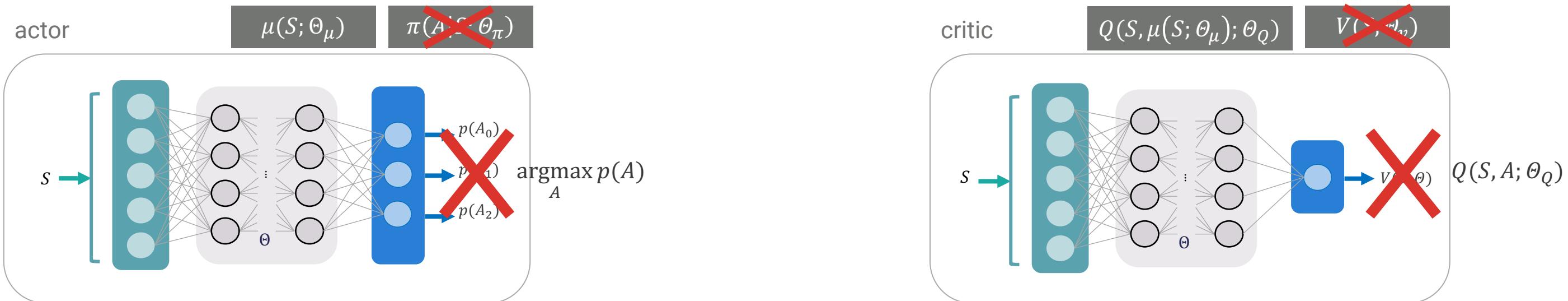


UNIVERSIDAD
DE GRANADA

Mejoras

Aproximaciones para espacios de acciones continuos

- *DDPG : Deep Deterministic Policy Gradient Continuous Action Space* (2015)



T. Lillicrap et al. (2015) *Continuous control with Deep Reinforcement Learning*. <https://arxiv.org/abs/1509.02971>

4 AlphaGo

CONCEPTOS FUNDAMENTALES

AlphaGo

Conceptos fundamentales



UNIVERSIDAD
DE GRANADA

Agente capaz de jugar al *Go* mejor que los humanos

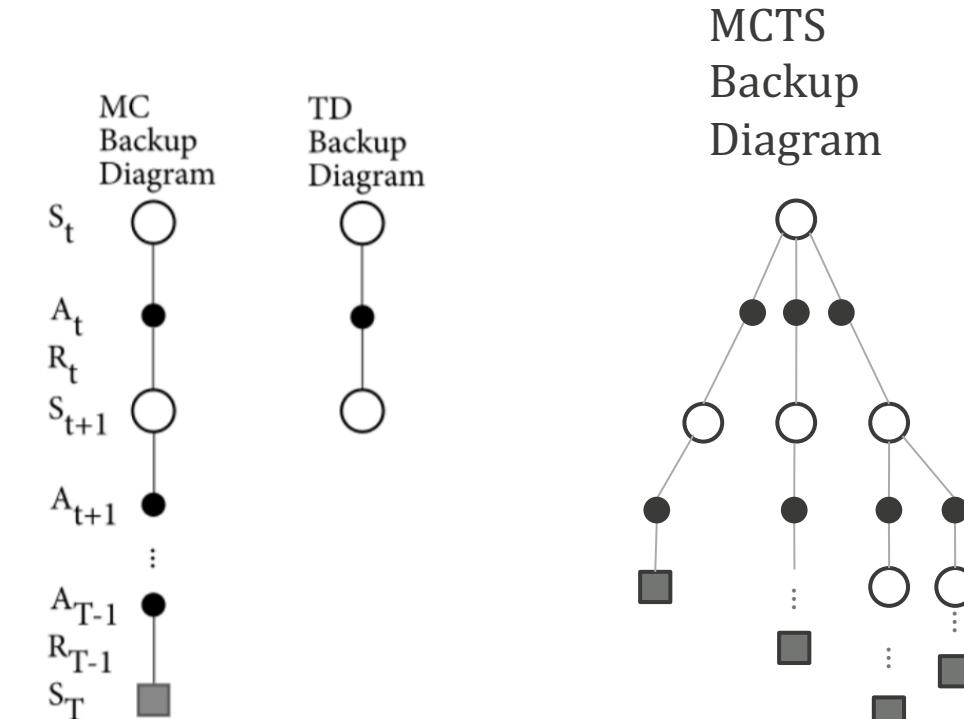
Aproximación

Resolución de juegos mediante árboles de búsqueda

Aprendizaje profundo por refuerzo

Método *actor-critic*

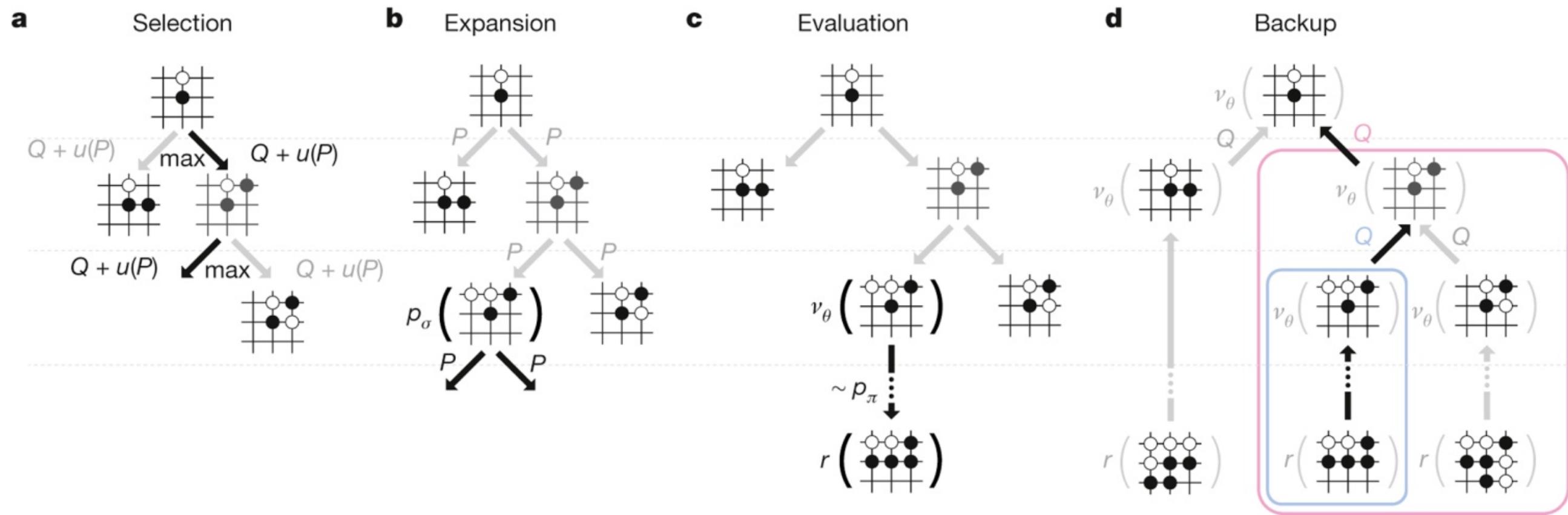
Rollout utilizando Monte Carlo Tree Search (MCTS)



D. Silver et al. (2016) *Mastering the game of Go with Deep Neural Networks & Tree Search*. Nature 529, 484-489

AlphaGo

Conceptos fundamentales



D. Silver et al. (2016) *Mastering the game of Go with Deep Neural Networks & Tree Search*. Nature 529, 484-489
M. Pumperla, K. Ferguson (2019) *Deep Learning and the Game of Go*. Manning.

AlphaGo

Versiones avanzadas



UNIVERSIDAD
DE GRANADA

AlphaZero (2017)

Aprende jugando contra sí mismo



D. Silver et al. (2017) *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play*. Science 362(6419), 1140-1144

AlphaStar (2019)

Es capaz de vencer a humanos en StarCraft 2.0



O. Vinyals et al. (2019) *AlphaStar: Mastering the Real-Time Strategy Game StarCraft II*. <https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/>

OpenAI DOTA 2 (2019)

Es capaz de vencer a humanos en DOTA 2



J. Pachocki et al. (2019) *OpenAI Five*. <https://openai.com/five/>

Pluribus (2019)

Es capaz de vencer a humanos en Poker



N. Brown, T. Sandholm (2019) *Superhuman AI for multiplayer poker*. Science 11 July 2019.

5 Aplicaciones

OTRAS

Aplicaciones

¿Qué más?

Implementaciones con bibliotecas disponibles públicamente:

- *Baselines* (OpenAI)
<https://github.com/openai/baselines>
- *RLlib: Scalable Reinforcement Learning*
<https://ray.readthedocs.io/en/latest/rllib.html>
- *ELF: An extensive, lightweight and flexible research platform for real-time strategy games*
<https://github.com/pytorch/elf>
- *Horizon RL* (Facebook)
<https://github.com/facebookresearch/Horizon>
- *TRFL* (Google DeepMind)
<https://github.com/deepmind/trfl/>
- *Huskarl* (Google Research)
<https://github.com/danaugrs/huskarl>

¡Y más! http://sl.ugr.es/drl_frameworks



UNIVERSIDAD
DE GRANADA

Retro game challenge, instrucciones
<https://contest.openai.com/2018-1/details/>



Aplicaciones

¿Qué más?



UNIVERSIDAD
DE GRANADA

Control eficiente de HVAC en edificios

- Utilización de un modelo de simulación del edificio para derivar control óptimo
- Problema de optimización combinatoria
 - Modelo de simulación simple → MILP
 - Modelo de simulación completo → Monte Carlo
 - ↑ Accuracy
 - ↑ Diversificación
 - ↑ Interpretabilidad
 - ↓ Eficiencia



J. Gomez-Romero et al. (2019) A Probabilistic Algorithm for Predictive Control With Full-Complexity Models in Non-Residential Buildings. IEEE Access 7, 38748 - 38765

Aplicaciones

¿Qué más?



UNIVERSIDAD
DE GRANADA

Aprendizaje (profundo) por refuerzo

Minimizar la energía consumida al tiempo que se mantiene la temperatura interior dentro de un rango de confort aplicando una secuencia de setpoints al equipamiento de climatización

> Modelar generación de planes como MDP (S, A, P_a, R_a, γ)

S : time, temperature

A : setpoint assignments

$$R_A = -\text{cost}(a_{t-1}, s_{t-1}) - \lambda \sum_{i=1}^z \left([\tau_t^i - \bar{\tau}_t^i]_+ + [\underline{\tau}_t^i - \tau_t^i]_+ \right) : \text{recompensa ponderada } (\lambda)$$

coste de la energía *violaciones de restricciones de temperatura*

$\gamma \in [0, 1]$: descuento

PROFICIENT: Deep Learning for Energy-Efficient Building Control

<https://jgromero.github.io/proficient/>

Ministerio de Ciencia, Innovación y Universidades – EXPLORA 2017, 11/2018-10/2020

IP: Juan Gómez-Romero



Índice

1. Deep Q-Learning
2. Métodos basados en política
3. Métodos *actor-critic*
4. AlphaGo
5. Aplicaciones