# Inteligencia Artificial

## CC-421

César Lara Avila

Universidad Nacional de Ingeniería

(actualización: 2021-01-23)

# Bienvenidos

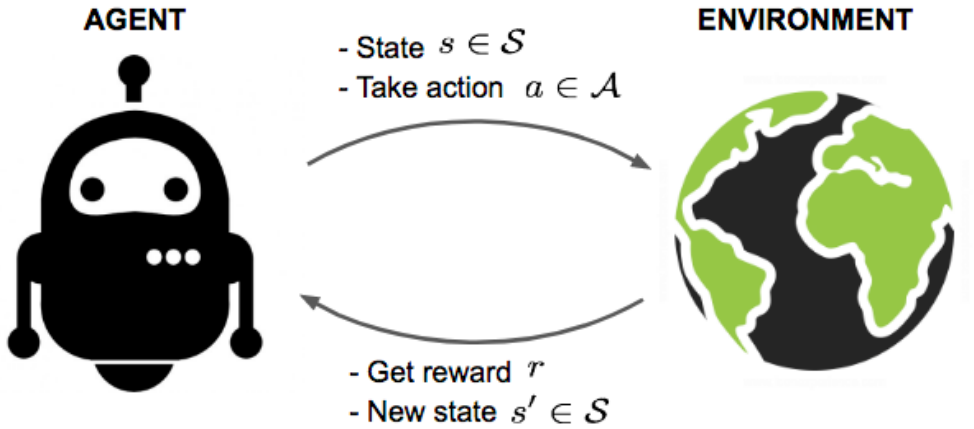
# Aprendizaje por Refuerzo - Parte 1

- ¿Qué es el aprendizaje por refuerzo?
- Conceptos y claves- terminología
- Políticas
- Problema del aprendizaje por Refuerzo
- Ecuaciones de Bellman
- Procesos de decisión de Markov

# ¿Qué es el aprendizaje por refuerzo?

El objetivo del aprendizaje por refuerzo (RL) es aprender una buena estrategia para un agente a partir de pruebas experimentales y la retroalimentación relativamente simple recibida.

Con la estrategia óptima, el agente es capaz de adaptarse activamente al entorno para maximizar las recompensas futuras.



# Aprendizaje por Refuerzo

- El Aprendizaje por Refuerzo (RL) es el problema abordado por un agente que tiene la tarea de aprender una conducta por medio de interacciones de prueba y error con un entorno dinámico. (Kaelbling 1996)
- Como en el caso de aprendizaje supervisado y no supervisado, es más una clase de problemas que un conjunto de técnicas.
- En cada interacción, el agente percibe una indicación del estado actual de su entorno y elige un acción.
- Esta acción altera el estado del entorno, cuyo valor es transmitido al agente como una señal de refuerzo.
- El agente tiene como tarea aprender las acciones que maximicen la suma total de las señales de refuerzo.

# Ejemplos de aprendizaje por Refuerzo

- Helicóptero autónomo con capacidad de hacer maniobras acrobáticas.
- Vencer al campión mundial en ajedrez
- Manejar un portafolio de inversiones
- Robot que aprende a caminar o jugar fútbol
- Jugar diferentes juegos de Atari mejor que los seres humanos.

# Conceptos y claves- terminología

Las partes principales de RL son el **agente** y el **entorno**. El entorno es el mundo en el que vive el agente y con el que interactúa. En cada paso de interacción, el agente ve una observación (posiblemente parcial) del estado del mundo y luego decide qué acción tomar.

El entorno cambia cuando el agente actúa sobre él, pero también puede cambiar por sí solo.

El agente también percibe una señal de **recompensa** del entorno, un número que le dice qué tan bueno o malo es el estado actual.

El objetivo del agente es maximizar la recompensa acumulativa, denominada **retorno**.

Los métodos de aprendizaje por refuerzo son formas en que el agente puede aprender comportamientos para lograr su objetivo.

# Exploración y explotación

El aprendizaje por refuerzo se basa en prueba y error.

El agente debe descubrir una buena política a partir de sus experiencias con el entorno sin perder mucha recompensa en el camino.

**Exploración** es buscar más información sobre el entorno.

**Explotación** es aprovechar la información conocida para maximizar la recompensa.

Usualmente es importante explorar y explotar.

**Ejemplos:**

- Explotación: Ir a tu restaurante favorito.
- Exploración: Probar un nuevo restaurante.
- Explotación: Mostrar el anuncio comercial más exitoso.
- Exploración: Mostrar un anuncio nuevo.

# Terminología

Para entender más específicamente de lo que hace RL, necesitamos introducir terminología adicional:

- Estados y observaciones
- Espacios de acción
- Políticas
- Trayectorias
- Diferentes formulaciones de retorno
- El problema de optimización de RL
- Funciones de valor.

# Estados y observaciones

Un **estado** $s$ es una descripción completa del estado del mundo. No hay información sobre el mundo que esté oculta al estado. Una observación $o$ es una descripción parcial de un estado, que puede omitir información.

En el RL profundo, casi siempre representamos estados y observaciones mediante un vector de valor real, una matriz o un tensor de orden superior.

Cuando el agente es capaz de observar el estado completo del entorno, decimos que el entorno está **completamente observado**.

Cuando el agente solo puede ver una observación parcial, decimos que el entorno se **observa parcialmente**.

# Espacios de acción

Los diferentes entornos permiten diferentes tipos de acciones. El conjunto de todas las acciones válidas en un entorno dado a menudo se denomina **espacio de acción**.

Algunos entornos, como Atari y Go, tienen espacios de acción discretos, donde el agente solo dispone de un número finito de movimientos.

Otros entornos, como donde el agente controla un robot en un mundo físico, tienen espacios de acción continua. En espacios continuos, las acciones son vectores de valor real.

Algunas familias de algoritmos solo se pueden aplicar directamente en un caso y tendrían que ser modificadas sustancialmente en el otro.

# Políticas

Una política es una regla que usa un agente para decidir qué acciones tomar. Nos dice qué acción tomar en el estado $s_t$.

Es un mapeo del estado $s_t$ a la acción $a_t$ y puede ser determinista o estocástica:

- Determinista: $a_t = \pi(s_t)$
- Estocástico: $a_t \sim \pi(\cdot|s_t) = \mathbb{P}_\pi[A = a_t|S = s_t]$.

En RL profundo, tratamos con **políticas parametrizadas**: políticas cuyas salidas son funciones calculables que dependen de un conjunto de parámetros que podemos ajustar para cambiar el comportamiento mediante algún algoritmo de optimización.

$$a_t = \pi_\theta(s_t)$$
$$a_t \sim \pi_\theta(\cdot|s_t).$$

# Políticas deterministas

Código para construir una política determinista simple para un espacio de acción continua en PyTorch, usando el paquete `torch.nn`:

```
pi_net = nn.Sequential(
nn. Lineal (obs_dim, 64),
nn.Tanh (),
nn Lineal (64, 64),
nn.Tanh (),
nn. Lineal (64, act_dim) )
```

Esto crea una red de perceptrón multicapa (MLP) con dos capas ocultas de tamaño $64$ y funciones de activación $\tanh$. Si $\mathrm{obs}$ es una matriz Numpy que contiene un lote de observaciones, $\mathrm{pi\_net}$ se puede usar para obtener un lote de acciones de la siguiente manera:

```
obs_tensor = torch.as_tensor (obs, dtype = torch.float32)
actions = pi_net (obs_tensor)
```

# Políticas estocásticas

Los dos tipos más comunes de políticas estocásticas en el RL profundo son las **políticas categóricas** y las **políticas Gaussianas diagonales**.

Las políticas categóricas se pueden usar en espacios de acción discretos, mientras que las políticas gaussianas diagonales se usan en espacios de acción continua.

Dos cálculos clave son de vital importancia para usar y entrenar políticas estocásticas:

- Muestreo de acciones de la política
- Cálculo de las probabilidades de registro de acciones particulares, $\log \pi_\theta(a|s)$.

### Políticas categóricas

Una política categórica es como un clasificador sobre acciones discretas.

Tu construyes una red neuronal para una política categórica de la misma manera que lo haría para un clasificador.

**Muestreo**. Dadas las probabilidades de cada acción, framework como PyTorch y Tensorflow tienen herramientas integradas para el muestreo.

Por ejemplo,en PyTorch se tiene, `torch.multinomial`.

**Log-verosimilitud**. Denota la última capa de probabilidades como $P_\theta(s)$. Es un vector con tantas entradas como acciones, por lo que podemos tratar las acciones como índices para el vector.

La probabilidad logarítmica de una acción a se puede obtener indexando en el vector:

$$\log \pi_\theta(a|s) = \log \left[P_\theta(s)\right]_a$$

## Políticas diagonales Gaussianas

Una distribución Gaussiana multivariada se describe mediante un vector medio, $\mu$, y una matriz de covarianza $\Sigma$.

Una política diagonal Gaussiana siempre tiene una red neuronal que mapea desde las observaciones hasta las acciones medias $\mu_\theta(s)$.

Hay dos formas diferentes en que se representa típicamente la matriz de covarianza.

**La primera forma:** hay un único vector de logarítmico de desviaciones estándar, $\log \sigma$ que no es una función del estado: $\log \sigma$ son parámetros independientes.

**La segunda forma:** hay una red neuronal que mapea los estados a desviaciones estándar logarítmicas, $\log \sigma_\theta(s)$.

**Muestreo** Dada la acción media $\mu_\theta(s)$ y la desviación estándar $\sigma_\theta(s)$ y un vector $z$ de ruido de un Gaussiano esférico, se puede calcular una muestra de acción con:

$$a = \mu_\theta(s) + \sigma_\theta(s) \odot z,$$

donde $\odot$ denota el producto elemento a nivel de dos vectores.

Los frameworks estándar tienen formas integradas de generar los vectores de ruido, como `torch.normal`.

Alternativamente, puedes construir objetos de distribución, por ejemplo, a través de `torch.distributions.Normal` y usarlos para generar muestras.

**Log-verosimilitud.** La probabilidad logarítmica de una acción k -dimensional $a$, para una diagonal Gaussiana con media $\mu = \mu_\theta(s)$ y desviación estándar $\sigma = \sigma_\theta(s)$, está dada por:

$$\log \pi_\theta(a|s) = -\frac{1}{2}\left(\sum_{i=1}^{k}\left(\frac{(a_i - \mu_i)^2}{\sigma_i^2} + 2\log\sigma_i\right) + k\log 2\pi\right).$$

# Trayectorias

Una trayectoria $\tau$ es una secuencia de estados y acciones en el mundo,

$$\tau = (s_0, a_0, s_1, a_1, \ldots)$$

El primer estado del mundo $s_0$, se muestrea aleatoriamente a partir de la distribución de estado inicial a veces denotado por $\rho_0$: $s_0 \sim \rho_0(\cdot)$.

Las transiciones de estado se rigen por las leyes naturales del entorno y dependen solo de las más reciente acción $a_t$.

Pueden ser deterministas,

$$s_{t+1} = f(s_t, a_t)$$

o estocástico,

$$s_{t+1} \sim P(\cdot|s_t, a_t).$$

Las acciones provienen de un agente de acuerdo con su política.

# Recompensa y retorno

La función de recompensa $R$ es de vital importancia en el aprendizaje por Refuerzo. Depende del estado actual del mundo, la acción que se acaba de tomar y el próximo estado del mundo:

$$r_t = R(s_t, a_t, s_{t+1})$$

A veces se puede escribir $r_t = R(s_t)$ o el par estado-acción $r_t = R(s_t, a_t)$.

El objetivo del agente es maximizar alguna noción de recompensa acumulativa a lo largo de una trayectoria. Anotaremos todos estos casos con $R(\tau)$.

Un tipo de retorno es el retorno sin descuento de horizonte finito, que es solo la suma de las recompensas obtenidas en una ventana fija de pasos:

$$R(\tau) = \sum_{t=0}^{T} r_t.$$

Otro tipo de retorno, es el retorno con descuento de horizonte infinito, que es la suma de todas las recompensas obtenidas por el agente, pero descontadas de qué tan lejos en el futuro se obtienen.

Esta formulación de recompensa incluye un factor de descuento $\gamma \in (0, 1)$:

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t.$$

**¿Por qué querríamos un factor de descuento? ¿No solo queremos obtener todas las recompensas?**.

# El problema del Aprendizaje por Refuerzo

El objetivo en RL es seleccionar una política que maximice el retorno esperado cuando el agente actúa de acuerdo con esa política.

Supongamos que tanto las transiciones del entorno como la política son estocásticas. En este caso, la la probabilidad de una trayectoria de paso $T$ es:

$$P(\tau|\pi) = \rho_0(s_0) \prod_{t=0}^{T-1} P(s_{t+1}|s_t, a_t)\pi(a_t|s_t).$$

El retorno esperado (para cualquier medida), denotado por $J(\pi)$, es entonces:

$$J(\pi) = \int_{\tau} P(\tau|\pi)R(\tau) = E_{\tau\sim\pi}[R(\tau)].$$

El problema central de optimización en RL puede entonces expresarse mediante:

$$\pi^* = \arg\max_{\pi} J(\pi),$$

siendo $\pi^*$ la política óptima.

# Funciones de valor

A menudo es útil conocer el valor de un estado o un par de estado-acción. Por **valor**, nos referimos al retorno esperado si comienza en ese estado o par estado-acción y luego actúa de acuerdo con una política particular para siempre.

Las **funciones de valor** se utilizan, de una forma u otra, en casi todos los algoritmos de RL.

Hay cuatro funciones principales a destacar aquí.

- La **función de valor en política**, $V^{\pi}(s)$, que da el retorno esperado si comienza en el estado $s$ y siempre actúa de acuerdo con la política $\pi$:

$$V^{\pi}(s) = E_{\tau \sim \pi}[R(\tau) \,|s_0 = s] \,.$$

- **La función de valor óptimo**, $V^{*}(s)$, que da el retorno esperado si comienza en el estado $s$ y siempre actúa de acuerdo con la política óptima en el entorno:

$$V^{*}(s) = \max_{\pi} E_{\tau \sim \pi}[R(\tau) \,|s_0 = s] \,.$$

- La **función de valor-acción en la política**, $Q^{\pi}(s, a)$, que da el retorno esperado si comienza en el estado $s$, toma una acción arbitraria $a$ (que puede no provenir de la política) y luego para siempre después de actuar de acuerdo con la política $\pi$:

$$Q^{\pi}(s, a) = E_{\tau \sim \pi}[R(\tau) \,|s_0 = s, a_0 = a] \,.$$

- La **función de valor-acción óptima**, $Q^{*}(s, a)$, que da el retorno esperado si comienza en el estado $s$, toma una acción arbitraria $a$ y luego actúa siempre de acuerdo con la política óptima en el entorno:

$$Q^{*}(s, a) = \max_{\pi} \; E_{\tau \sim \pi}[R(\tau) \,|s_0 = s, a_0 = a] \,.$$

Hay dos conexiones clave entre la función de valor y la función de valor-acción que surgen con bastante frecuencia:

$$V^{\pi}(s) = E_{a \sim \pi}[Q^{\pi}(s, a)],$$

$$V^{*}(s) = \max_{a} Q^{*}(s, a).$$

# La función Q-óptima y la acción óptima

Existe una conexión importante entre la función de valor de acción óptima $Q^*(s, a)$ y la acción seleccionada por la política óptima.

Por definición, $Q^*(s, a)$ da el retorno esperado para comenzar en el estado $s$, tomar la acción (arbitraria) $a$ y luego actuar de acuerdo con la política óptima para siempre.

La política óptima en $s$ seleccionará la acción que maximice el retorno esperado a partir de $s$.

Como resultado, si tenemos $Q^*$, podemos obtener directamente la acción óptima, $a^*(s)$, a través de:

$$a^*(s) = \arg\max_a Q^*(s, a).$$

# Ecuaciones de Bellman

Las cuatro funciones de valor obedecen a ecuaciones especiales de autoconsistencia llamadas **ecuaciones de Bellman**.

La idea básica detrás de las ecuaciones de Bellman es la siguiente:

El valor de tu punto de partida es la recompensa que esperas obtener por estar allí, más el valor del lugar donde llegas a continuación.

Las ecuaciones de Bellman para las funciones de valor de política son

$$V^{\pi}(s) = \underset{\substack{a\sim\pi \\ s'\sim P}}{E}\left[r(s,a) + \gamma V^{\pi}(s')\right],$$

$$Q^{\pi}(s,a) = E_{s'\sim P}\left[r(s,a) + \gamma E_{a'\sim\pi}[Q^{\pi}(s',a')]\right],$$

donde $s' \sim P$ es la abreviatura de $s' \sim P(\cdot|s,a)$, que indica que el siguiente estado $s'$ se extrae de las reglas de transición del entorno. $a \sim \pi$ es la abreviatura de $a \sim \pi(\cdot|s)$ y $a' \sim \pi$ es la abreviatura de $a' \sim \pi(\cdot|s')$.

Las ecuaciones de Bellman para las funciones de valor óptimo son:

$$V^*(s) = \max_a E_{s' \sim P}[r(s,a) + \gamma V^*(s')],$$
$$Q^*(s,a) = E_{s' \sim P}[r(s,a) + \gamma \max_{a'} Q^*(s',a')].$$

El término **Bellman backup** para un estado, o par estado-acción, es el lado derecho de la ecuación de Bellman: la recompensa más el siguiente valor.

# Funciones de ventaja

En RL, no necesitamos describir qué tan buena es una acción en un sentido absoluto, sino solo cuánto mejor es en promedio que otras. Es decir, queremos conocer la ventaja relativa de esa acción.

Hacemos este concepto preciso con la **función de ventaja**.

La función de ventaja $A^{\pi}(s, a)$ correspondiente a una política $\pi$ describe cuánto mejor es realizar una acción específica $a$ en el estado $s$, sobre la selección aleatoria de una acción de acuerdo con $\pi(\cdot|s)$, asumiendo que actúas de acuerdo con $\pi$ para siempre.

Matemáticamente, la función de ventaja se define como:

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s).$$

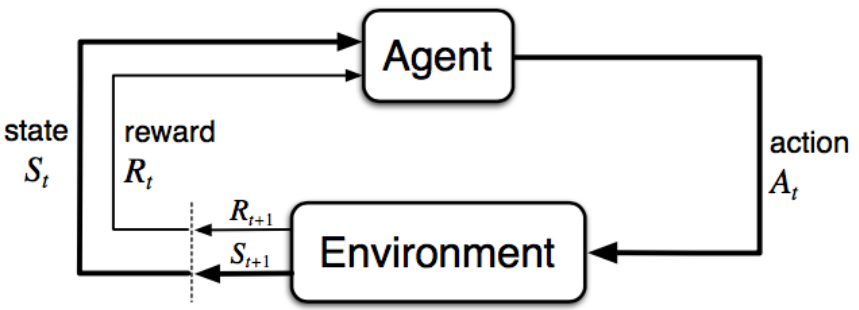
# Procesos de decisión de Markov

En términos más formales, casi todos los problemas de RL pueden enmarcarse como Procesos de Decisión de Markov (MDP).

Todos los estados en MDP tienen la **propiedad Markov**, refiriéndose al hecho de que el futuro solo depende del estado actual, no de la historia:

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, \ldots, S_t]$$

O en otras palabras, el futuro y el pasado son condicionalmente independientes dado el presente, ya que el estado actual encapsula todas las estadísticas que necesitamos para decidir el futuro (Sutton-Barto).



Un proceso de decisión de Markov consta de cinco elementos $\mathcal{M} = \langle S, A, P, R, \rho_0 \rangle$ donde:

- $S$ es el conjunto de estados
- $A$ es el conjunto de todas las acciones válidas
- $P : S \times A \to P(S)$ es la función de probabilidad de transición con $P(s^{'}|s, a)$ la probabilidad de hacer la transición al estado $s'$ si comienzas en el estado $s$ y realizas la acción $a$.
- $R : S \times A \times S \to \mathbf{R}$ es una función de recompensa con $r_t = R(s_t, a_t, s_{t+1})$,
- $\rho_0$ es la distribución del estado inicial.

En un entorno desconocido, no tenemos un conocimiento perfecto sobre $P$ y $R$.

# Fin!