| Inteligencia Artificial - Aprendizaje por Refuerzo

Microsoft

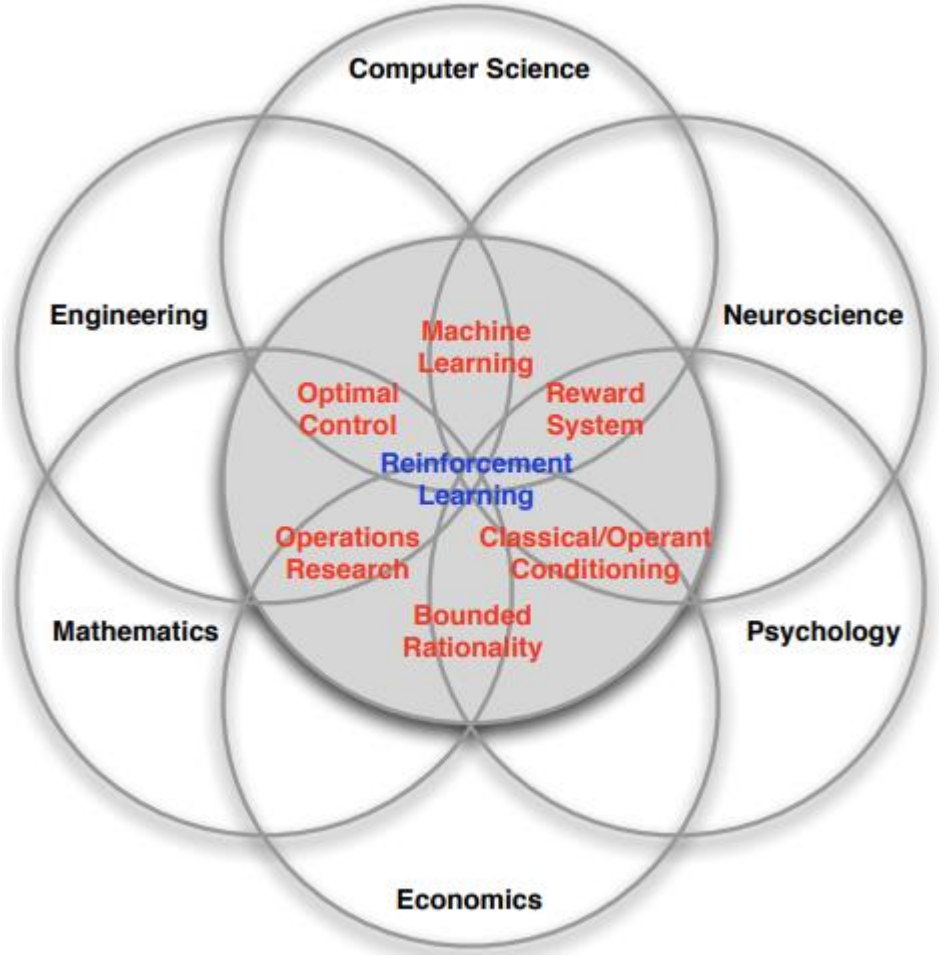
Aprendizaje por refuerzo

- Aprendemos a través de la interacción con nuestro entorno
- Cuando aprendemos, muchas veces no hay supervisor explícito, pero sí hay una conexión **sensoriomotora** directa con el entorno.
- El ejercicio de esta conexión produce una gran cantidad de información acerca de la causa y el efecto, sobre las consecuencias de las acciones, y acerca de qué hacer con el fin de lograr los objetivos.

Aprendizaje por refuerzo

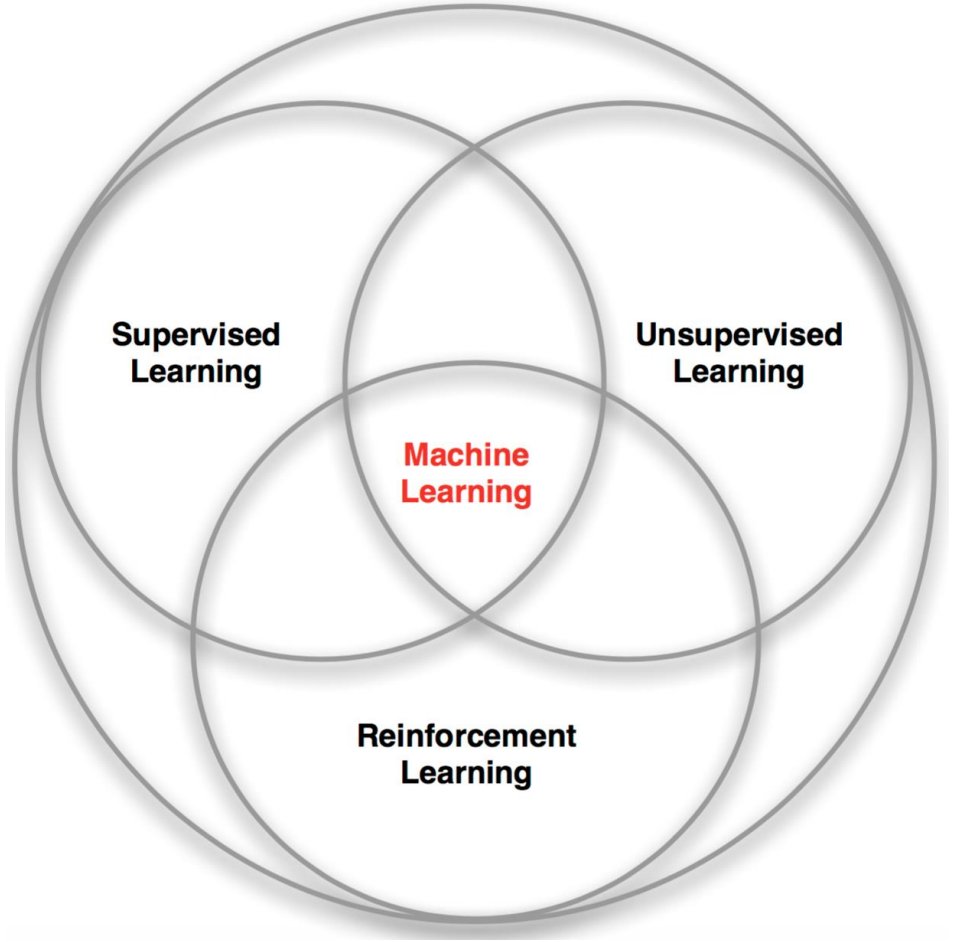
- ¿Qué hace que el aprendizaje por refuerzo sea diferente de otros paradigmas de aprendizaje de máquina?
  - No hay ningún supervisor, solamente una señal de recompensa
  - La retroalimentación no es instantánea
  - El tiempo realmente importa (secuencial, observaciones no son independientes e idénticamente distribuidas)
    - Las acciones del agente afectan a los datos posteriores que recibe

Aprendizaje por refuerzo



- Con el aprendizaje por refuerzo estudiamos la forma óptima de tomar decisiones

Tipos de aprendizaje máquina



Aprendizaje por refuerzo

- Es el aprendizaje de qué hacer - cómo mapear situaciones a acciones - con el fin de maximizar una señal de recompensa numérica.
- Al aprendiz no se le dice qué acciones tomar, como en la mayoría de las formas de aprendizaje de la máquina, sino que debe descubrir qué acciones producen la mayor recompensa probándolas.

Ejemplos

- Realizar maniobras acrobáticas en un helicóptero
- Derrotar al campeón del mundo en Backgammon
- Administrar una cartera de inversión
- Controlar una central eléctrica
- Hacer que un robot humanoide camine
- Jugar muchos juegos de Atari diferentes mejor que los seres humanos

Juegos resueltos con inteligencia artificial

Juego	Nivel de juego	Programa
Damas Americanas	Perfecto	Chinook
Ajedrez	Súper-humano	Deep Blue
Otelo	Súper-humano	Logistello
Backgammon	Súper-humano	TD-Gammon
Scrabble	Súper-humano	Maven
Go	Gran maestro	MoGo[1], Crazy Stone[2], Zen y AlphaGo[3]
Poker[4]	Súper-humano	Polaris

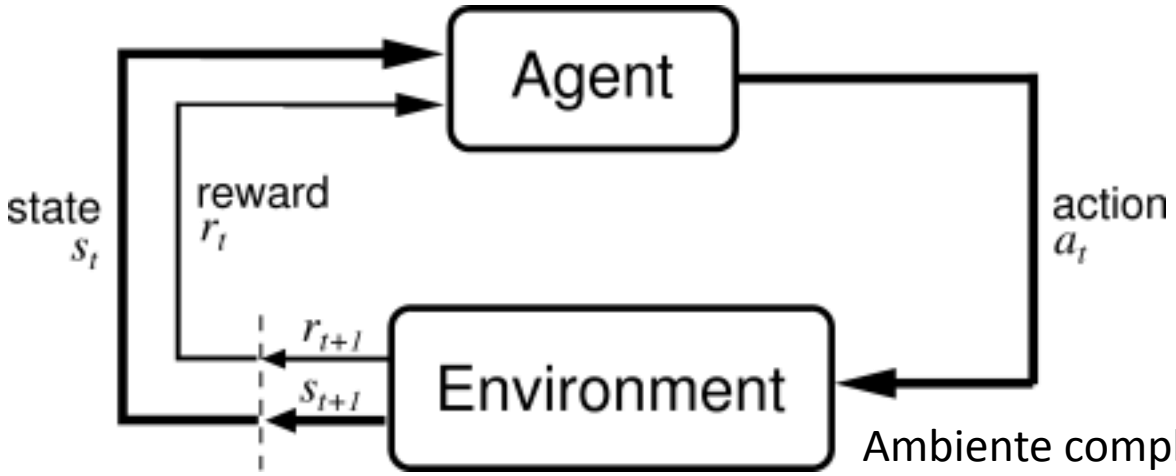
1 9 x 9
2 9 x 9 y 19 x 19
3 19 x 19
4 Heads-up Limit Texas Hold'em

Juegos resueltos con aprendizaje por refuerzo

Juego	Nivel de juego	Programa
Damas Americanas	Perfecto	Chinook
Ajedrez	Maestro internacional	KnightCap / Meep
Otelo	Súper-humano	Logistello
Backgammon	Súper-humano	TD-Gammon
Scrabble	Súper-humano	Maven
Go	Gran maestro	MoGo[1], Crazy Stone[2], Zen y AlphaGo[3]
Poker[4]	Súper-humano	SmooCT

1 9 x 9
2 9 x 9 y 19 x 19
3 19 x 19
4 Heads-up Limit Texas Hold'em

Agente



Ambiente completamente observable

$r_{t+1} \in \Re$ Recompensa numérica

$s_t \in \mathcal{S}$ Posibles estados

$a_t \in \mathcal{A}(s_t)$ Posibles acciones

$t = 0, 1, 2, 3, \ldots$

$\pi_t(s, a)$ Política: probabilidad de seleccionar la acción ***a*** si el estado es ***s*** en el tiempo ***t***

El objetivo del agente es maximizar el monto del refuerzo que recibe a largo plazo

Recompensa

- Una recompensa R_t es una señal de realimentación escalar
- Indica que tan bien se ha comportado el agente en el tiempo t
- El objetivo del agente es maximizar la acumulación de recompensas
- El aprendizaje de refuerzo se basa en la hipótesis de recompensa
  - Todos los objetivos se pueden describir a través de la maximización de la recompensa acumulada esperada

Ejemplos de recompensas

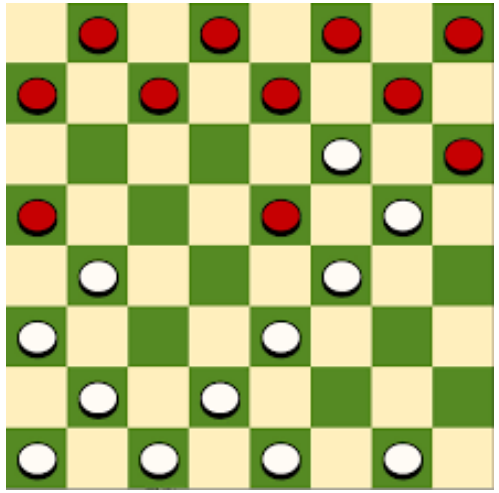
- Realizar maniobras acrobáticas en un helicóptero
  - Recompensa positiva por seguir la trayectoria deseada
  - Recompensa negativa por desplomarse
- Derrotar al campeón del mundo en Backgammon
  - Recompensa positiva por ganar
  - Recompensa negativa por perder
- Administrar una cartera de inversión
  - Recompensa positiva por cada $ ganado
- Controlar una central eléctrica
  - Recompensa positiva por la generación de energía
  - Recompensa negativa por exceder los umbrales de seguridad
- Hacer que un robot humanoide camine
  - Recompensa positiva por avanzar
  - Recompensa negativa por caerse
- Jugar muchos juegos de Atari diferentes mejor que los seres humanos
  - Recompensa positiva por incrementar la puntación
  - Recompensa negativa por disminuir la puntuación

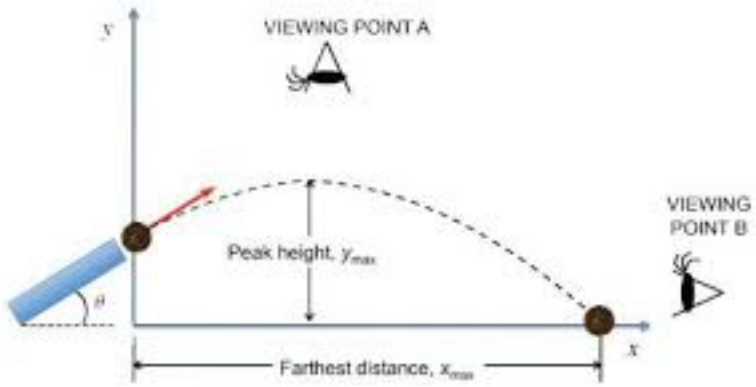
Secuencia de toma de decisiones

- **Objetivo**: seleccionar acciones para maximizar la recompensa futura total
- Las acciones pueden tener consecuencias a largo plazo
- La recompensa puede retrasarse
- Puede ser mejor sacrificar recompensa inmediata para ganar más recompensas a largo plazo
- Ejemplos:
  - Una inversión financiera (puede tardar meses para madurar)
  - Cargar combustible a un helicóptero (podría prevenir un accidente en varias horas)
  - Bloquear los movimientos de un oponente (podría mejorar las posibilidades de ganar dentro de muchos movimientos a partir de ahora)

Estados

- El agente toma sus decisiones en función de una señal del entorno que le llamamos estado
- Una señal de estado que logra retener toda la información relevante se dice que es Markov

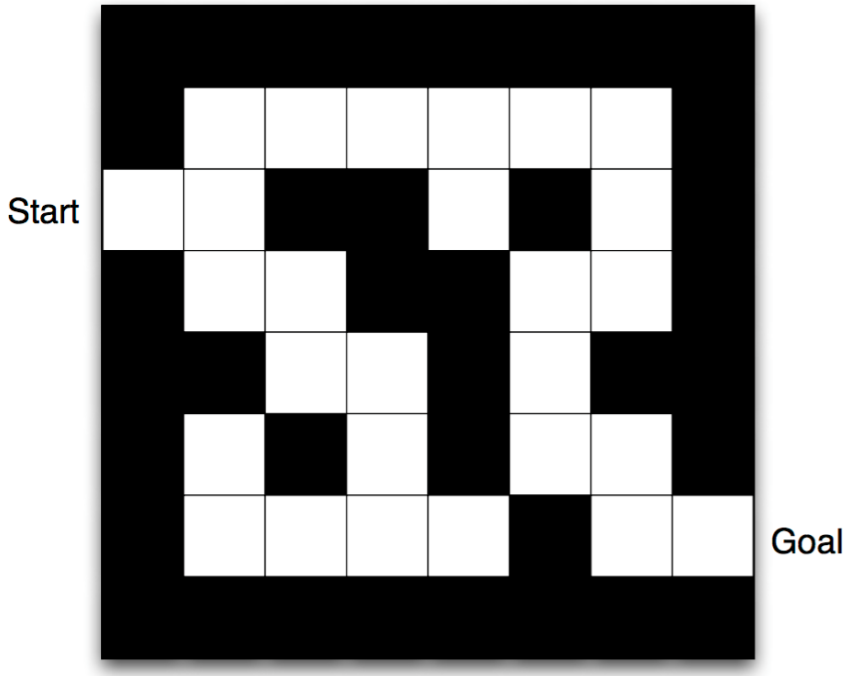




Propiedad de Markov

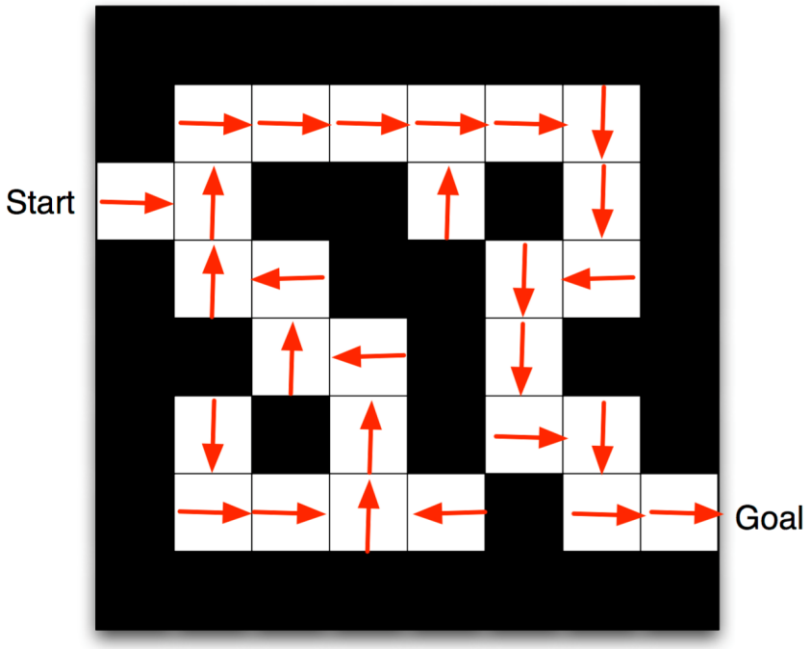
- El futuro es independiente del pasado dado el presente
- Una vez que el estado es conocido, el pasado puede descartarse

Ejemplo del laberinto



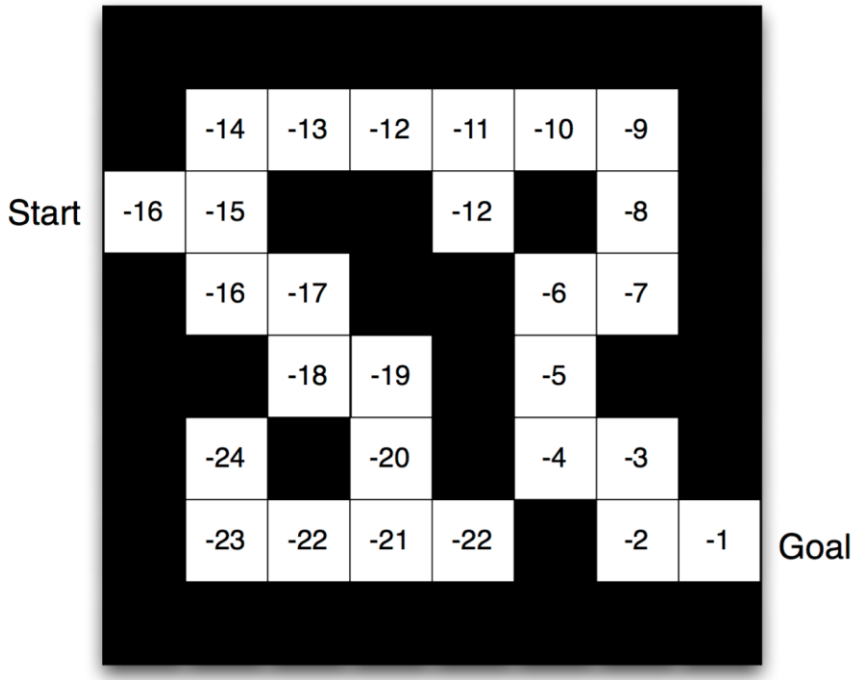
- **Recompensa:** -1 por cada paso dado
- **Acciones:** N, S, E, O
- **Estados:** Ubicación del agente

Ejemplo del laberinto: Política



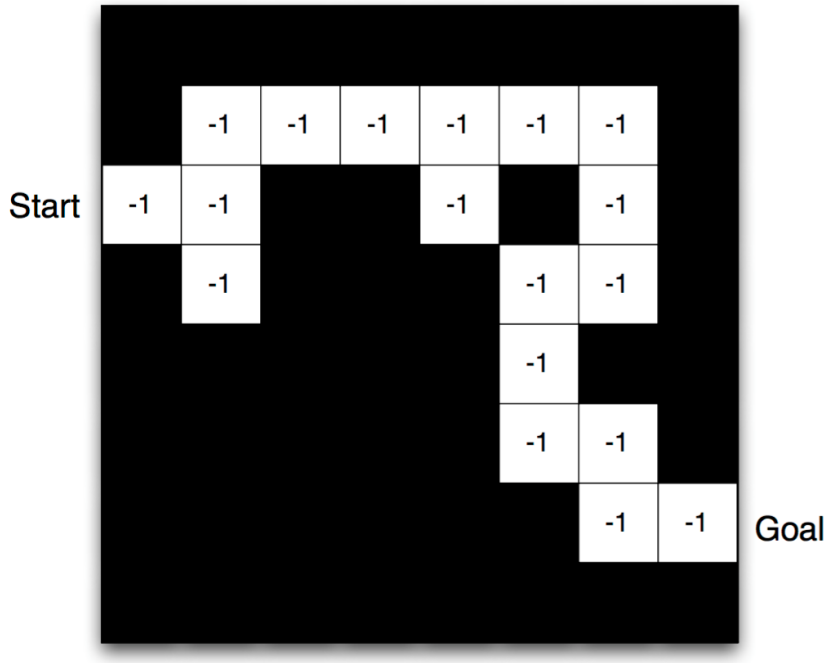
- Las flechas representan la política π(s) para cada estado s

Ejemplo del laberinto: Función de valor



- Número representa el valor $V_\pi(s)$ de cada estado s

Ejemplo del laberinto: Modelo



$\mathcal{P}^a_{ss'}$ Cuadrícula

$\mathcal{R}^a_s$ Números

- Los agentes pueden tener un modelo del ambiente
- Dinámica: como las acciones cambian el ambiente
- Recompensas: cuanta por cada estado
- El modelo puede se imperfecto

Tipo de agentes AR

- Basados en Valores
  - Sin política (implícita)
  - Funciones de valor
- Basados en Política
  - Política
  - Sin función de valor
- Actor - Crítico
  - Política
  - Funciones de valor

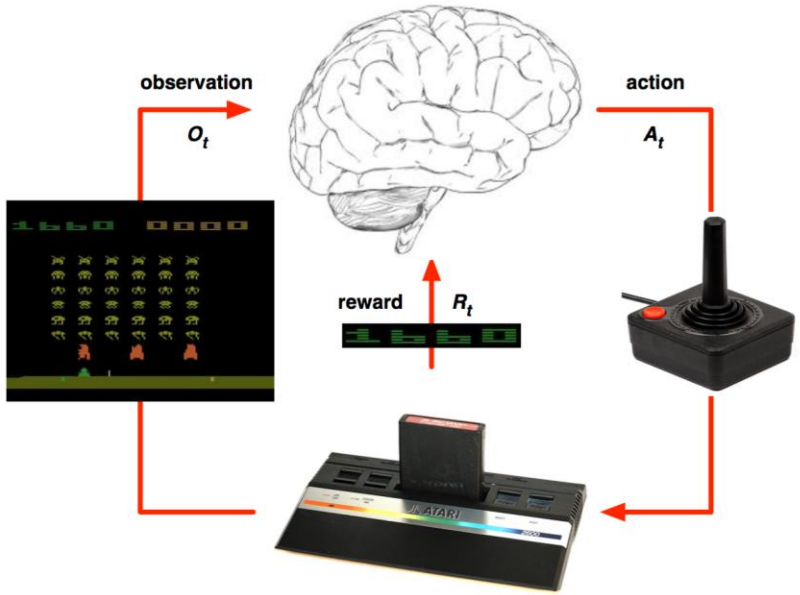
Tipo de agentes AR

- Libre de modelos
  - Política y/o funciones de valor
  - Sin modelo
- Basados en modelos
  - Política y/o funciones de valor
  - Modelo

Aprendizaje y planificación

- Dos problemas fundamentales en la secuencia de toma de decisiones
  - Aprendizaje por refuerzo
    - El ambiente es inicialmente desconocido
    - El agente interactúa con el ambiente
    - El agente mejora su política
  - Planificación
    - Un modelo del ambiente es conocido
    - El agente realiza cálculos con su ambiente (sin interacción externa)
    - El agente mejora su política
    - Conocido como deliberación, razonamiento, introspección, reflexión, pensamiento, búsqueda

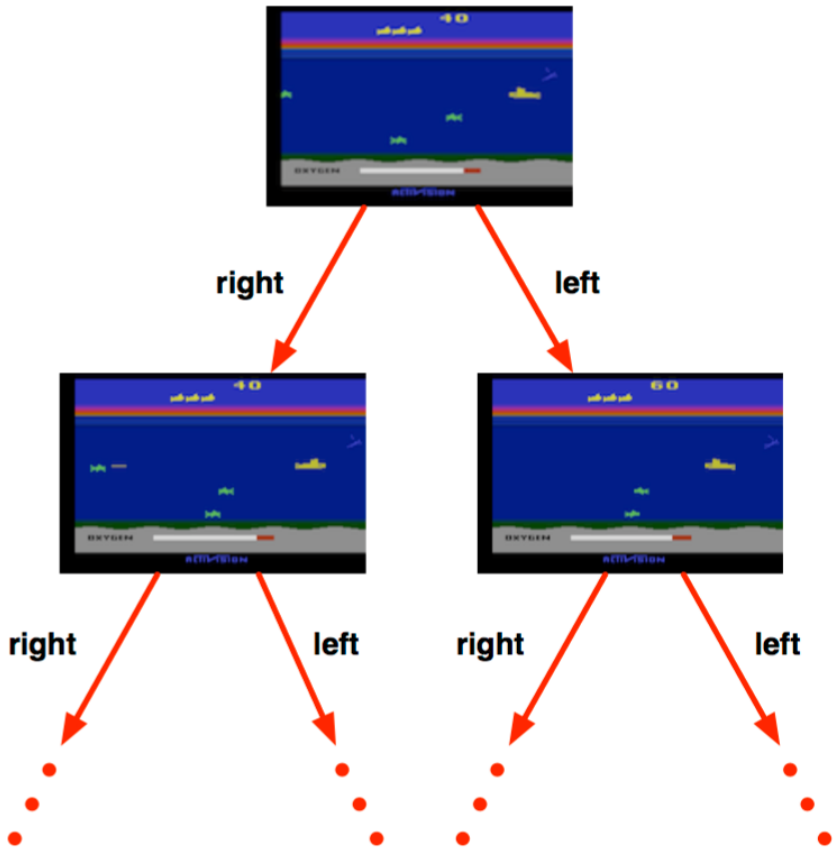
Atari: Aprendizaje por refuerzo



- Reglas de los juegos son desconocidos
- Se aprende directamente jugando
- Se seleccionan acciones a realizar con el joystick, se observa la imagen y la puntuación

Atari: Planificación

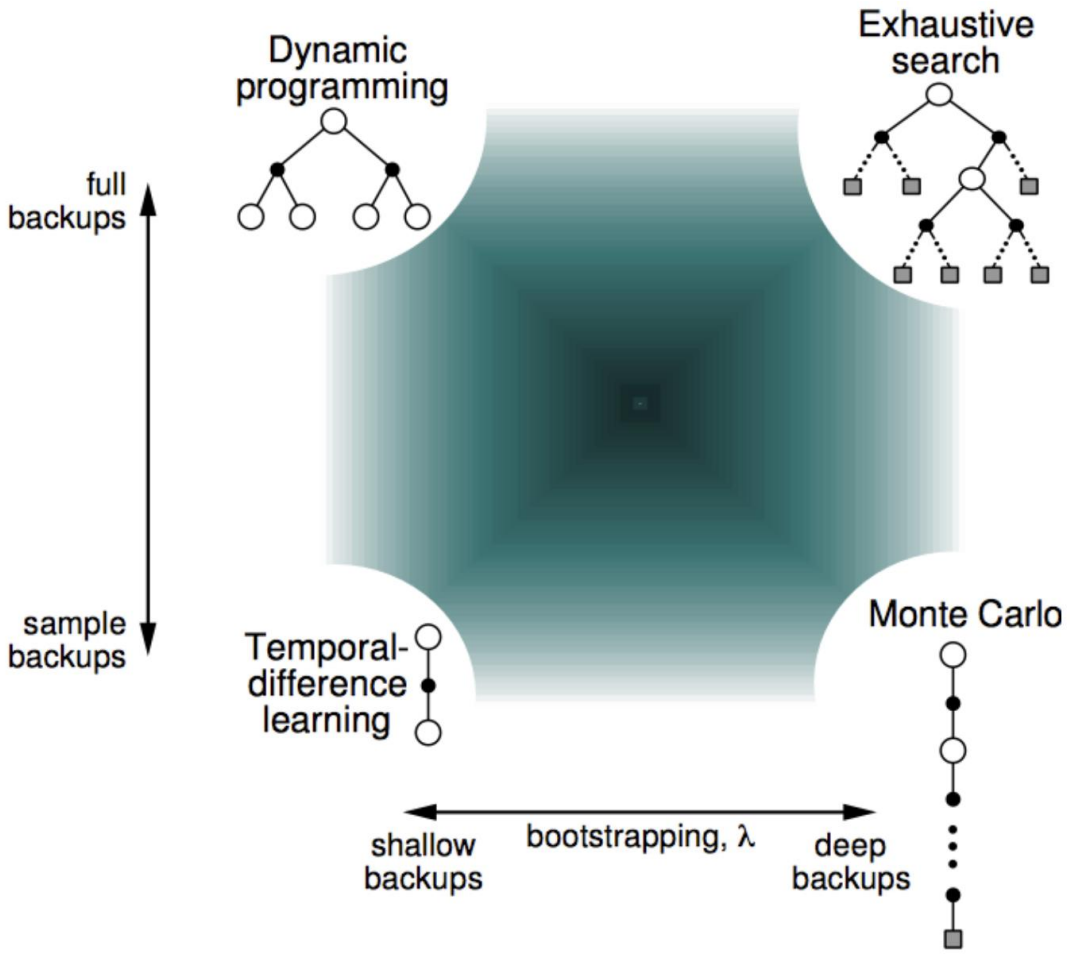
- Reglas del juego son conocidas
- Podemos acceder al emulador
  - Modelo perfecto dentro del agente
  - Puedo saber el estado y puntuación que obtendré seleccionando una acción
- Planifico con antelación para obtener política optima
  - Un árbol de búsqueda



Exploración y explotación

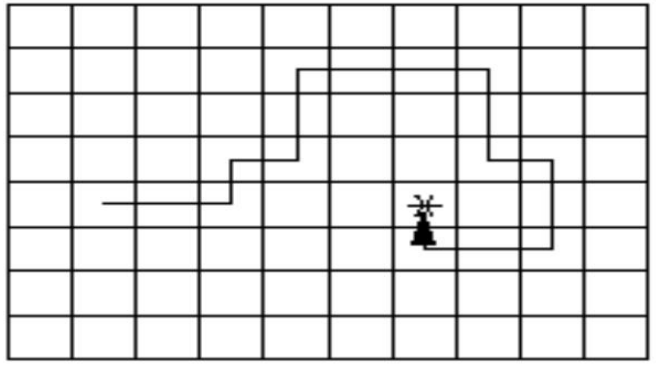
- Uno de los retos es el mantener el equilibrio entre la **exploración** y **explotación** en este método.
  - Para maximizar la recompensa, un agente de aprendizaje por refuerzo debe preferir las acciones que ha intentado en el pasado y han sido efectivas en la obtención de recompensa.
  - Para descubrir este tipo de acciones, tiene que intentar acciones que no se ha seleccionado antes.

Vista unificada de aprendizaje por refuerzo

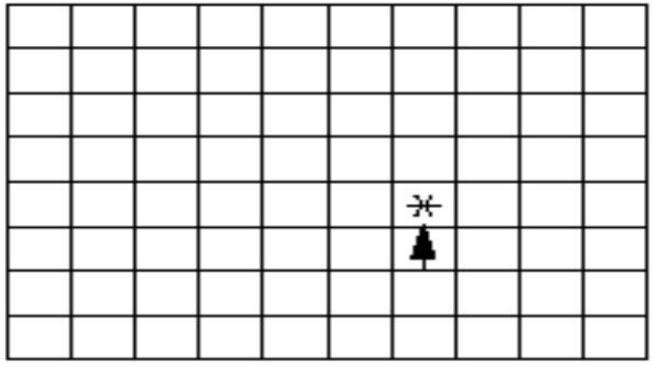


Ejemplo mundo cuadriculado Sarsa(λ)

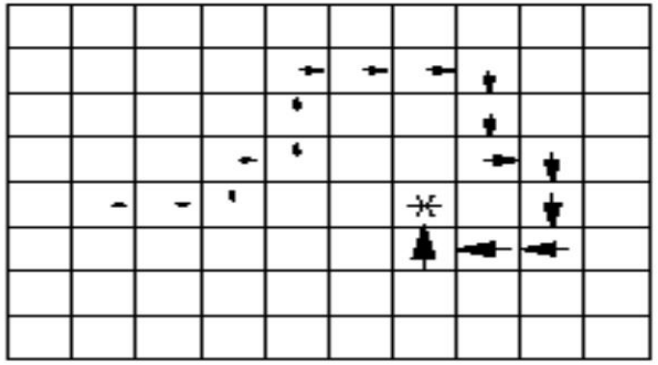
Path taken



Action values increased by one-step Sarsa



Action values increased by Sarsa(λ) with λ=0.9



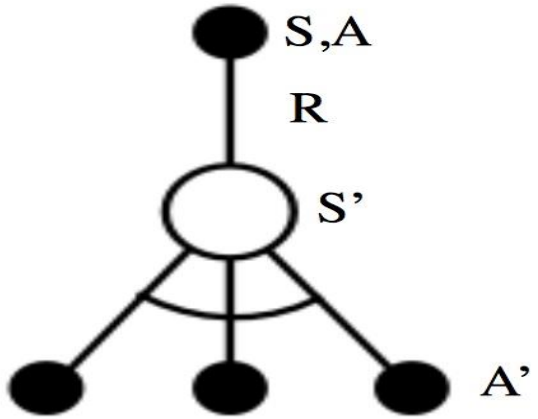
Aprendizaje fuera de política

- Evaluar la política de destino $\pi(a|s)$ para calcular $v_\pi(s)$ o $q_\pi(s, a)$
- Mientras se sigue la política de comportamiento $\mu(a|s)$

$$\{S_1, A_1, R_2, ..., S_T\} \sim \mu$$

- ¿Porque es esto importante?
  - Aprender de la observación de seres humanos u otros agentes
  - Reutilizar la experiencia generada a partir de políticas antiguas $\pi_1, \pi_2, ..., \pi_{t-1}$
  - Aprender la política óptima mientras sigue una política exploratoria
  - Aprender varias políticas mientras sigue una política

Control Q learning



$$Q(S,A) \leftarrow Q(S,A) + \alpha \left(R + \gamma \max_{a'} Q(S',a') - Q(S,A) \right)$$

Algoritmos de control incremental

- Para MC, el objetivo es G_t

$$\Delta \mathbf{w} = \alpha(G_t - \hat{q}(S_t, A_t, \mathbf{w}))\nabla_{\mathbf{w}}\hat{q}(S_t, A_t, \mathbf{w})$$

- Para TD(0), el objetivo es $R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$

$$\Delta \mathbf{w} = \alpha(R_{t+1} + \gamma\hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}) - \hat{q}(S_t, A_t, \mathbf{w}))\nabla_{\mathbf{w}}\hat{q}(S_t, A_t, \mathbf{w})$$

- Para TD(λ) hacia adelante, el objetivo es q_t^{λ}

- Para T

$$\Delta \mathbf{w} = \alpha(q_t^{\lambda} - \hat{q}(S_t, A_t, \mathbf{w}))\nabla_{\mathbf{w}}\hat{q}(S_t, A_t, \mathbf{w})$$

$$\delta_t = R_{t+1} + \gamma\hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}) - \hat{q}(S_t, A_t, \mathbf{w})$$
$$E_t = \gamma\lambda E_{t-1} + \nabla_{\mathbf{w}}\hat{q}(S_t, A_t, \mathbf{w})$$
$$\Delta \mathbf{w} = \alpha\delta_t E_t$$

Deep Q-Networks (DQN)

DQN uses experience replay and fixed Q-targets

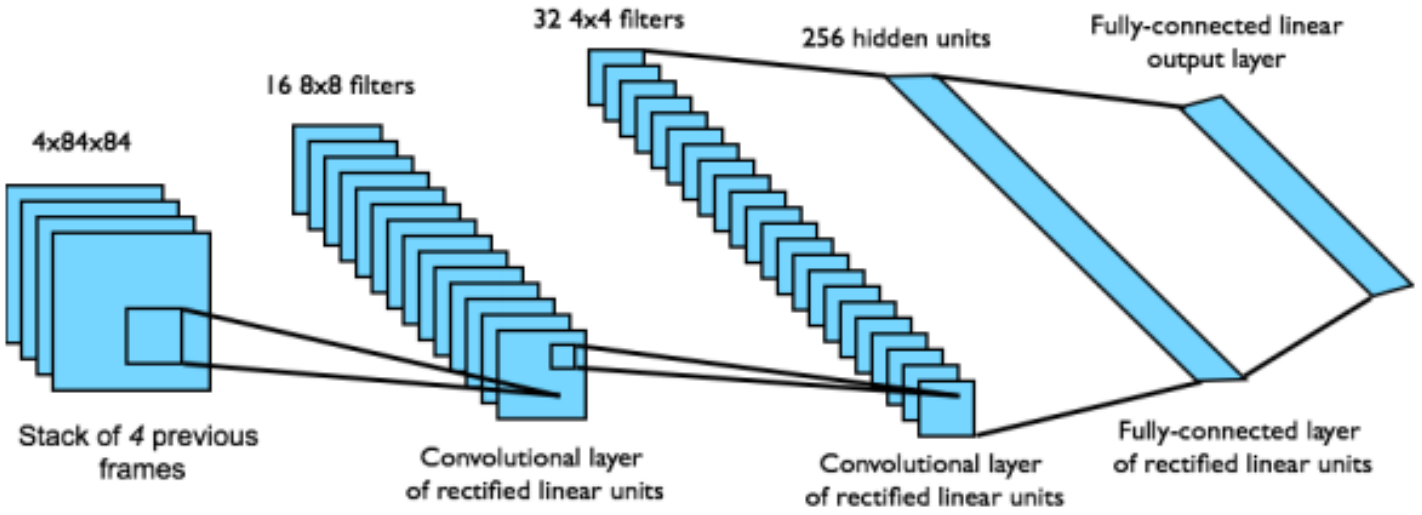
- Take action a_t according to ϵ-greedy policy
- Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in replay memory $\mathcal{D}$
- Sample random mini-batch of transitions (s, a, r, s') from $\mathcal{D}$
- Compute Q-learning targets w.r.t. old, fixed parameters w^-
- Optimise MSE between Q-network and Q-learning targets

$$\mathcal{L}_i(w_i) = \mathbb{E}_{s,a,r,s' \sim \mathcal{D}_i}\left[\left(r + \gamma \max_{a'} Q(s', a'; w_i^-) - Q(s, a; w_i)\right)^2\right]$$

- Using variant of stochastic gradient descent

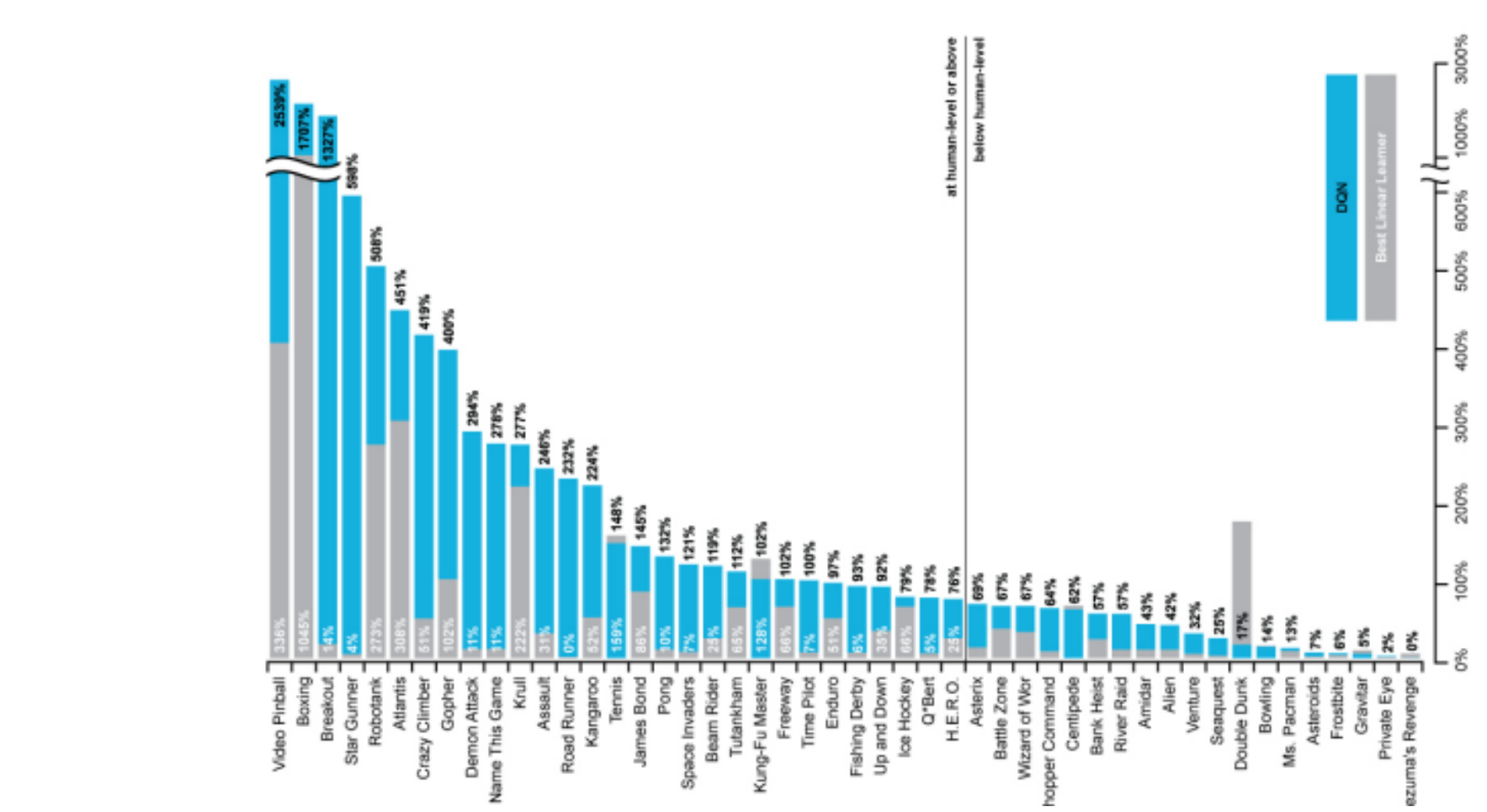
DQN en Atari

- End-to-end learning of values $Q(s, a)$ from pixels s
- Input state s is stack of raw pixels from last 4 frames
- Output is $Q(s, a)$ for 18 joystick/button positions
- Reward is change in score for that step



Network architecture and hyperparameters fixed across all games

Resultados de DQN en Atari



Microsoft