2016-2017

Aprendizaje Automático

2. Aprendizaje computacional: límites y paradigmas

Enrique Vidal Ruiz
(evidal@dsic.upv.es)

Francisco Casacuberta Nolla
(fcn@dsic.upv.es)

Departament de Sistemas Informàtics i Computació (DSIC)

Universitat Politècnica de València (UPV)

Septiembre, 2016

Index

Planteamiento formal de aprendizaje inductivo

En *aprendizaje inductivo* el sistema posee escaso conocimiento a-priori sobre la tarea a a resolver y obtiene su(s) modelo(s) principalmente mediante *ejemplos* o muestras de *entrada/salida* de dicha tarea.

En un planteamiento formal intervienen los siguientes elementos:

- *Método o algoritmo de aprendizaje,* $\mathcal{A}$
- *Clase* $\mathcal{G}$ *de funciones a aproximar o "aprender"*. Toda $g \in \mathcal{G}$ es de la forma $g\colon \mathcal{X} \to \mathcal{Y}$. Para cada tarea, se asume que existe alguna $g \in \mathcal{G}$ que la representa exactamente[1]
- *Clase* $\mathcal{F}$ *funciones con las que se representan los "modelos" resultado del aprendizaje*. Toda $f \in \mathcal{F}$ es de la forma: $f\colon \mathcal{X} \to \mathcal{Y}$
- *Muestra finita de aprendizaje* $S \subset \mathcal{X} \times g(\mathcal{X})$
- *Modo de presentación de la muestra*. Indica cómo se extraen las muestras S de $\mathcal{X} \times g(\mathcal{X})$
- *Criterio de éxito*. Indica qué se espera de $\mathcal{A}$ al final del aprendizaje

1. Esta definición obvia la existencia de una *función de representación* que asigna a cada objeto real un elemento del *espacio de representación*, $\mathcal{X}$. Cuando esto se tiene en cuenta, en general $\mathcal{G}$ no puede ser un espacio de *funciones*, sino de *relaciones* de la forma: $\mathcal{G} \subset \mathcal{X} \times \mathcal{Y}$.

Criterios de éxito

Criterio de *identificación en el límite*: comportamiendo de un método de AA para tallas crecientes de los conjuntos de aprendizaje. Este criterio de éxito ha sido ampliamente utilizado en el campo de la *Inferencia Gramatical* (IG). Conclusiones computacionales pesimistas. En general, AA considera este criterio poco útil en la práctica.

Criterio de *aprendizaje probablemente aproximadamente correcto* (PAC): el algoritmo AA debe selecionar con alta probabilidad un modelo cuyo error de generalización sea pequeño.

Otros criterios más realistas desde un punto de vista práctico se plantean en el *Marco estadístico* de AA.

Aprendizaje probablemente aproximadamente correcto (PAC)

El aprendizaje PAC [Valiant,84] trata de relacionar:

- la clase de funciones a aproximar (aprender), $\mathcal{G}$, y su complejidad
- la clase de modelos o funciones resultado del aprendizaje, $\mathcal{F}$
- la forma en la que se presentan los datos de entrenamiento
- el número de datos de entrenamiento necesarios
- la precisión con la que se aproxima cualquier $g \in \mathcal{G}$
- la probabilidad de que un método de AA, $\mathcal{A}$, aprenda con éxito
- el coste computacional del aprendizaje

Las funciones a aprender, $\mathcal{G}$, son de la forma $g : \mathcal{X} \rightarrow \mathcal{Y}$ (análogamente para $\mathcal{F}$) aunque, por simplicidad, se suele asumir que las salidas, $y \in \mathcal{Y}$, son binarias; es decir, $\mathcal{Y} \equiv \mathbb{B}$.

Definición de clase PAC-aprendible

Sean $\mathcal{G}$ y $\mathcal{F}$ clases de funciones. Sea $g \in \mathcal{G}$ una funcion a aproximar mediante el método de AA $\mathcal{A}$ y denotemos con $|g|$ la "talla" (de la descripción) de g.

Sea $\mathcal{D}$ una distribución de probabilidad sobre los datos de $\mathcal{X}$ y sea $E_{\mathcal{D}}(\text{error}(f, g))$ la esperanza estadística (según $\mathcal{D}$) del error de aproximar g mediante $f \in \mathcal{F}$.

Se dice que $\mathcal{G}$ es PAC-aprendible mediante $\mathcal{A}$ si:

$\forall g \in \mathcal{G}, \quad \forall \mathcal{D}, \quad \forall \epsilon, 0 < \epsilon < 1/2, \quad \forall \delta, 0 < \delta < 1/2,$

dada una muestra de g de talla N extraida según $\mathcal{D}$, con probabilidad mayor o igual que $(1 - \delta)$ y con tiempo de cómputo polinómico en $1/\epsilon, 1/\delta, N, |g|$ $\mathcal{A}$ obtiene $f \in \mathcal{F}$ tal que $E_{\mathcal{D}}(\text{error}(f, g)) \leq \epsilon$.

El paradigma PAC tiene gran interés teórico, aunque su aplicación práctica es bastante limitada.

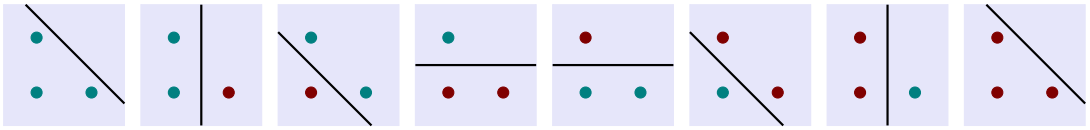
Dimensión de Vapnik-Chervonenkis (VC)

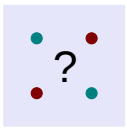
La dimensión VC mide de la capacidad de *separación* de una clase de funciones de *clasificación*, $\mathcal{G} = \{g \colon \mathcal{X} \to \{1, \ldots, C\}\}$.

Se dice que $\mathcal{G}$ *separa* a un conjunto finito de datos $S \subset \mathcal{X}$ si, para cualquier asignación de etiquetas de clase a los elementos de S, existe una $g \in \mathcal{G}$ que clasifica sin error todos los elementos de S.

La dimensión VC de $\mathcal{G}$, $h(\mathcal{G})$, es la máxima cardinalidad de un conjunto $S \subset \mathcal{X}$ separable por $\mathcal{G}$.

Ejemplo:
$\mathcal{X} = \mathbb{R}^2$, $C = 2$, $\mathcal{G} \equiv$ funciones lineales en $\mathbb{R}^2$. Según la figura, $h(\mathcal{G}) = 3$.





Límites computacionales del aprendizaje PAC

Sea e_f el error empírico de una función $f \in \mathcal{F}$ medido sobre el conjunto, S, cuya talla es N. En PAC se pueden demostrar las siguientes cotas generales:

- Si la clase de funciones a aprender es finita ($|\mathcal{G}| < \infty$):
  - Mínimo N, necesario para aprendizaje PAC:
$$N \geq \frac{1}{\epsilon}(\ln |\mathcal{G}| + \ln \frac{1}{\delta})$$
  - Mínimo error esperado ("de *test*"):
$$\max_g E_{\mathcal{D}}(\text{error}(f, g)) \leq e_f + \sqrt{\frac{\ln |\mathcal{G}| + \ln \frac{1}{\delta}}{2N}}$$
- Si $|\mathcal{G}| = \infty$, sea $h = h(\mathcal{G})$ la dimensión VC de $\mathcal{G}$:
  - $$N \geq \frac{4}{\epsilon}(2h \log \frac{13}{\epsilon} + \log \frac{2}{\delta})$$
  - $$\max_g E_{\mathcal{D}}(\text{error}(f, g)) \leq e_f + \sqrt{\frac{h(\log \frac{2N}{h} + 1) + \log \frac{4}{\delta}}{N}}$$

Index

Teoría de la decisión estadística

- La "función"[1] a aprender, g, es arbitraria (es decir, no se hace ninguna asunción sobre la clase $\mathcal{G}$)
- El modelo que se aprende, $f\colon \mathcal{X} \to \mathcal{Y}$, se considera una *"función de decisión"*
- Para definir un *criterio de éxito* (simplificado, pero útil en la práctica) se asume que, para cada $x \in \mathcal{X}$, la *"decisión"* $f(x)$ solo puede ser *"acertada"* o *"errónea"* (por ej., según $f(x)$ sea idéntica o bastante similar a $g(x)$, o no)
- Toda decisión tiene un *coste*. El planteamiento más simple asume que si la decisión $f(x)$ es *acertada* su coste es 0 y si es *errónea* su coste es 1
- La función de decisión, f, se basa en la *probabilidad a posteriori*, $P(y \mid x)$, estimada a partir una muestra de entrenamiento $S \subset \mathcal{X} \times g(\mathcal{X})$
- El *criterio de éxito* es minimizar la esperanza estadística del coste de las decisiones sobre todos los posibles datos de entrada $x \in \mathcal{X}$. Con la simplificación de coste $0/1$, esto equivale a minimizar la probabilidad de error
- *Este planteamiento es la base del marco estadístico en AA y RF*

1. En este caso, g *no* es una *función* propiamente hablando, sino una *relación* de la forma $g :\subset \mathcal{X} \times \mathcal{Y}$

Teoría de la decisión estadística: minimizar el riesgo de error

Sea $x \in \mathcal{X}$ un dato de *entrada* y sea $y \in \mathcal{Y}$ una *decisión* que se toma para x. $P(y \mid x)$ representa la probabilidad de que la decisón y sea correcta.

Probabilidad de error si se toma la decisión y:

$$P_y(\text{error} \mid x) = 1 - P(y \mid x)$$

Mínima probabilidad de error:

$$\forall x \in \mathcal{X}: \quad P_\star(\text{error} \mid x) = \min_{y \in \mathcal{Y}} P_y(\text{error} \mid x) \;=\; 1 - \max_{y \in \mathcal{Y}} P(y \mid x)$$

Para cada x la probabilidad de error se minimiza si se toma la decisión con mayor $P(y \mid x)$; o sea, la decisión que es "probablemente más acertada").

Mínimo riesgo global o mínima esperanza de error de decisión:

$$P_\star(\text{error}) \;=\; \int_{x \in \mathcal{X}} P_\star(\text{error}, x)\, dx \;=\; \int_{x \in \mathcal{X}} P_\star(\text{error} \mid x)\, P(x)\, dx$$

Función de decisión de mínimo riesgo de error o de Bayes:

$$\forall x \in \mathcal{X}: \quad f^\star(x) = \arg\max_{y \in \mathcal{Y}} P(y \mid x)$$

Marco estadístico de AA

- *Aprendizaje*: Estimación de $P(y \mid x)$ mediante algún criterio adecuado
- *Decisión* (clasificación o regresión): Para cada dato de *test*, $x \in \mathcal{X}$, calcular $f^\star(x)$; es decir, obtener una y tal que $P(y \mid x)$ sea máxima.

Criterio de máxima verosimilitud (MV) para estimación de $P(y \mid x)$:
Se asume que la función de probabilidad conjunta $P(x, y)$ depende de un *vector de parámetros*[1], $\boldsymbol{\theta}$; es decir, $P(x, y) \equiv P(x, y \mid \boldsymbol{\theta}), \boldsymbol{\theta} \in \mathbb{R}^D$.

Dado un conjunto de entrenamiento $S \subset \mathcal{X} \times \mathcal{Y}$, la probabilidad (o *"verosimilitud"*) de que $P(x, y \mid \boldsymbol{\theta})$ genere S, y su logaritmo, son:

$$p(S \mid \boldsymbol{\theta}) \;=\; \prod_{(x,y) \in S} P(x, y \mid \boldsymbol{\theta})\,, \qquad L_S(\boldsymbol{\theta}) \;=\; \sum_{(x,y) \in S} \log P(x, y \mid \boldsymbol{\theta})$$

Estimación de máxima verosimilitud de $\boldsymbol{\theta}$:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L_S(\boldsymbol{\theta})$$

MV es consistente con la minimización de la esperanza del error de decisión.

1. Ej: (μ, σ) de una Gausiana, o probs. de transición y emisión en modelos ocultos de Markov discretos.

Marco estadístico de AA: regla de Bayes

Frecuentemente puede simplificarse el aprendizaje por MV descomponiendo las probabilidades conjunta y/o a posteriori, mediante la *regla de Bayes*:

$$P(y \mid x) \;=\; \frac{P(x,y)}{P(x)} \;=\; \frac{P(y)\; P(x \mid y)}{P(x)}$$

De esta forma, la función de decisión resulta:

$$f^\star(x) = \arg\max_y P(y)\; P(x \mid y)$$

donde $P(y)$ es la *probabilidad a priori* de y y $P(x \mid y)$ es la *probabilidad condicional* de x dada y.

Análogamente, la log-verosimilitud se descompone como:

$$L_S(\boldsymbol{\theta}) \;=\; \sum_{(x,y)\in S} \log P(x,y \mid \boldsymbol{\theta}) \;=\; \sum_{(x,y)\in S} \log P(y \mid \boldsymbol{\theta}) \;+\; \sum_{(x,y)\in S} \log P(x \mid y, \boldsymbol{\theta})$$

lo que permite estimar por separado (por MV) $P(y \mid \boldsymbol{\theta})$ y $P(x \mid y, \boldsymbol{\theta})$.

Index

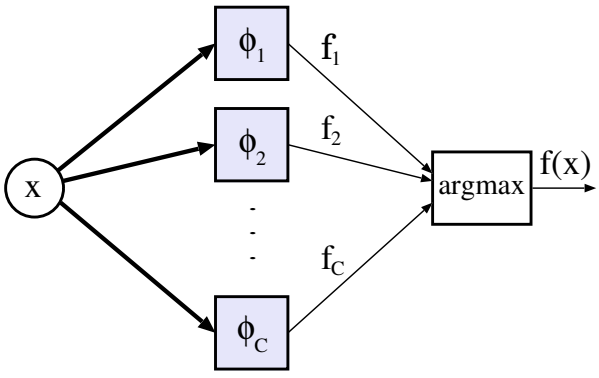
Planteamiento conexionista o basado en funciones discriminantes

- Puede considerarse que corresponde a la aproximación al RF, tradicionalmente llamada *"no paramétrica"* (o *"no estadística"*)
- No se require ninguna asunción probabilística sobre el dominio de las funciones a aprender, $\mathcal{X} \times \mathcal{Y}$
- El *criterio de éxito* se establece directamente en términos de minimización del error de decisión sobre un *conjunto de test*
- La clase de modelos $\mathcal{F}$ se suele definir mediante *funciones discriminantes* (FD); típicamente funciones *lineales* o *generalizaciones* (en algunos casos llamadas *"núcleos"*)
- Las funciones de decisión del marco estadístico del AA pueden verse como un caso particular de FDs y, a la inversa, las FDs entrenadas bajo ciertos criterios son aproximaciones a las probabilidades a posteriori

Funciones discriminantes y clasificación

Todo *clasificador* $f : \mathcal{X} \to \{1, \ldots, C\}$ puede expresarse mediante C *funciones discriminantes*, $\phi_1, \phi_2, \ldots, \phi_C$, $\phi_c : \mathcal{X} \to \mathbb{R}$, $1 \leq c \leq C$:

$$f(x) = \underset{1 \leq c \leq C}{\arg\max}\ \phi_c(x)$$



Por ejemplo, en el marco estadístico, $\phi_c(x) \equiv P(c \mid x)$.

En el caso de *regresión*, si $\mathcal{Y} = \mathbb{R}^{d'}$, se puede hacer $C = d'$ y eliminar la maximización ($\arg\max$) (o bien $C \geq d'$ sustituyendo $\arg\max$ por alguna *función de combinación* adecuada $F : \mathbb{R}^C \to \mathbb{R}^{d'}$).

Index

Otras aproximaciones al AA

- AA basado en *proximidad*: aprendizaje por *memorización* basado en una medida de *disimilitud o distancia* y en almacenar prototipos.
- Computación *evolutiva* y *algoritmos genéticos*: técnicas de aprendizaje inspiradas en ideas de selcción natural en biología.
- *Aprendizaje por refuerzo*: Un agente debe aprender una *política óptima* de interacción con su entorno para conseguir un *objetivo*.
- *Aprendizaje no-supervisado*. Solo se disponen de datos de entrada no etiquetados. Se prentende encontrar agrupaciones “naturales” de datos.
- *Aprendizaje semi-supervisado*. Solo una parte de los datos de aprendizaje están etiquetados.
- *Aprendizaje activo*. El algoritmo de aprendizaje determina cuántos y cuales de los datos de entrada (de entrenamiento) deben ser etiquetados por un operador humano.

AA basado en *proximidad*

- Se asume que $\mathcal{X}$ es un espacio *métrico* o al menos *pseudométrico* (aunque no necesariamente vectorial)
- Se define una medida de *disimilitud o distancia* adecuada, $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, de manera que si dos objetos x, x' son *"similares"*, $d(x, x')$ es pequeña
- El aprendizaje es básicamente por *memorización*: los *prototipos* son directamente los propios datos de entrenamiento, S, o subconjuntos adecuadamente seleccionados de S
- En problemas de clasificación, las *clases* se representan mediante (conjuntos de) *prototipos*; es decir, elementos (adecuadamente escogidos) de $\mathcal{X}$ cuya etiqueta de clase es conocida. La función de clasificación f, se basa en las distancias entre un dato de *test* y los prototipos de las distintas clases
- Existen extensiones adecuadas para problemas de *regresión*
- Este planteamiento de AA se encuadra propiamente en el marco estadístico considerando que la probabilidad a posteriori $P(c \mid x)$ se estima mediante las distancias de x a prototipos de c

Computación evolutiva y algoritmos genéticos

El aprendizaje "evolutivo" o "genético" puede considerarse una rama de la llamada *computación evolutiva o genética*.

Computación evolutiva (CE):

- Estudia sistemas computacionales inspirados en ideas (darwinianas) de *selección natural*
- Entre estas ideas, destacan la *mutación* y la *supervivencia del más apto*
- Las técnicas de CE son aplicables a diversos campos: optimización, diseño, AA, etc.
- Las técnicas de CE no requieren un rico conocimiento del dominio aunque, para que sean efectivas, generalmente es preciso incorporar algo (¡o mucho!) de este conocimiento en el diseño de los algoritmos

Un esquema simple de algoritmo genético

1. ***Generar*** aleatoriamente una *población inicial* (de soluciones), $B(0)$; $i \leftarrow 0$

2. ***REPETIR:***

   (a) ***Evaluar*** la *aptitud* $A(Z)$ de cada solución $Z \in B(i)$

   (b) ***Seleccionar*** aleatoriamente 2 (o más) *padres*, $Z, Z' \in B(i)$ tales que $A(Z)$ y $A(Z')$ sean grandes

   (c) ***Generar*** n *descendientes*, $Z_1, \ldots, Z_n$, mediante operaciones aleatorias de *cruce* entre Z y Z' y *mutación* del resultado de los *cruces*

   (d) $i \leftarrow i + 1$; $B(i) \leftarrow Z_1, \ldots, Z_n$

3. ***HASTA*** que se satisfaga una *condición de FIN*

Algoritmo evolutivo: ejemplo

Maximizar $f : \mathbb{N} \to \mathbb{R}$, $f(x) = -x^2 + 30x + 30$, en el intervalo $0 \leq x \leq 31$

- *Individuos*: números naturales $x \in [0, \ldots .31]$
- *Codificación* de un individuo: en binario (a esto se le llama *"cromosoma"*)
- *Población inicial*, por ejemplo, 4 individuos: $B(0) = \{01101, 11000, 01000, 10011\}$
- *Aptitud*: $A(01101) = f(13) = 251$, $A(11000) = f(24) = 174$,
  . $A(01000) = f(8) = 206$, $A(10011) = f(19) = 239$
- *Selección* de $n = 2$ *padres*: aleatoria, según A. Probabilidad de selección: $P(Z) = A(Z)/\sum_{Z'' \in B(i)} A(Z'')$; supongamos que salen: $Z = 10011$ y $Z' = 01000$
- *Cruce*: Seleccionamos posiciones de corte entre 0 y 5. Por ejemplo, para $n = 4$, generamos 2 posiciones aleatorias de corte; supongamos que salen 2 y 4: *Cruce*$(10011, 01000) = \{10\,000, 01\,011,\ 1001\,0, 0100\,1\}$
- *Mutación*: Un método simple es cambio aleatorio de bits (con baja probabilidad). Un resultado podría ser: $B(1) = \{10000, 01\mathbf{1}11, 10010, 0\mathbf{0}001\}$
- Entre la población $B(1)$ está la solución: $\hat{Z} = 01111$, $f(\hat{Z}) = 255$. Pero el algoritmo debe seguir hasta que se cumpla la *condición de fin*; por ejemplo, hasta que la mejor *aptitud* no varíe significativamente entre dos generaciones

Algoritmos evolutivos y AA

Se han aplicado con cierto éxito a problemas simples de aprendizaje *supervisado* (inductivo), *no-supervisado* y *por refuerzo* en el campo de la *inteligencia artificial*:

- Aprendizaje de sistemas (de clasificación) *basados en reglas*
- Aprendizaje de *redes neuronales* "evolutivas"
- Aprendizaje de sistemas de *lógica difusa*
- Aprendizaje por *co-evolución* (competición)
- etc.

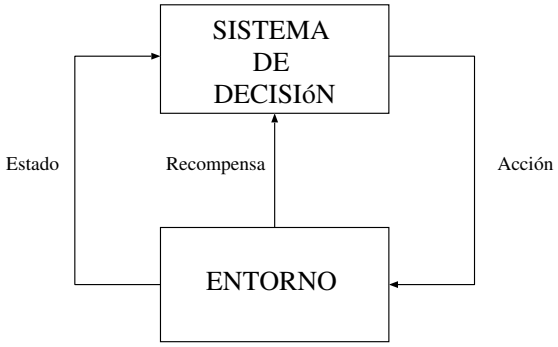
El interés para problemas reales prácticos es limitado.

Aprendizaje por refuerzo (AR)

- En AR, un agente (es decir un sistema) *interactúa* con el entorno para conseguir un *objetivo*. Mediante la propia actividad de interacción, el agente debe *aprender una política de interacción*; es decir, una estrategia de elección de posibles *acciones*
  - La realización de una acción cambia el *estado* del entorno y las consecuencias de ello son sancionadas por el entorno mediante una señal de *recompensa* (premio o castigo)
  - Cada posible *estado* del entorno tiene un *valor* relacionado con la esperanza de la recompensa acumulada o *ganancia* (*"return"*) que se puede lograr a partir de ese estado, siguiendo la política que se está aprendiendo
  - En esencia, el aprendizaje en AR consiste en obtener *estimaciones* adecuadas *de los valores* de los estados.
- AR presenta un compromiso entre *explotación* de lo aprendido hasta el momento y *exploración* de acciones no realizadas previamente
- Ejemplos: Jugar al ajedrez, control de varios ascensores, vehículos autónomos, robots, etc.

Elementos en aprendizaje por refuerzo

- Conjunto de *estados* $\mathcal{Q}$
- Conjunto de *acciones* $\mathcal{A}$
- Probabilidad de *transición (Markoviana)*: $\tau(q' \mid q, a),\ q, q' \in \mathcal{Q},\ a \in \mathcal{A}$
- Función de *recompensa* $r : \mathcal{Q} \times \mathcal{A} \times \mathcal{Q} \to \mathbb{R}$



- En el paso t el estado es $q \in \mathcal{Q}$. El agente escoge una acción $a \in \mathcal{A}$ y se alcanza un nuevo estado $q' \in \mathcal{Q}$ (según $\tau(Q_{t+1} = q' \mid Q_t = q, A_t = a)$), con una recompensa $R_t = r(q, a, q')$
- La *ganancia* o recompensa acumulada para una secuencia de pasos (con "horizonte infinito") es:

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

donde γ ($0 \leq \gamma < 1$) es un *factor de descuento o amortiguamiento*

Ejemplo: robot de reciclado

Modelo de proceso de decisión markoviano de un robot de recogida de latas vacías:

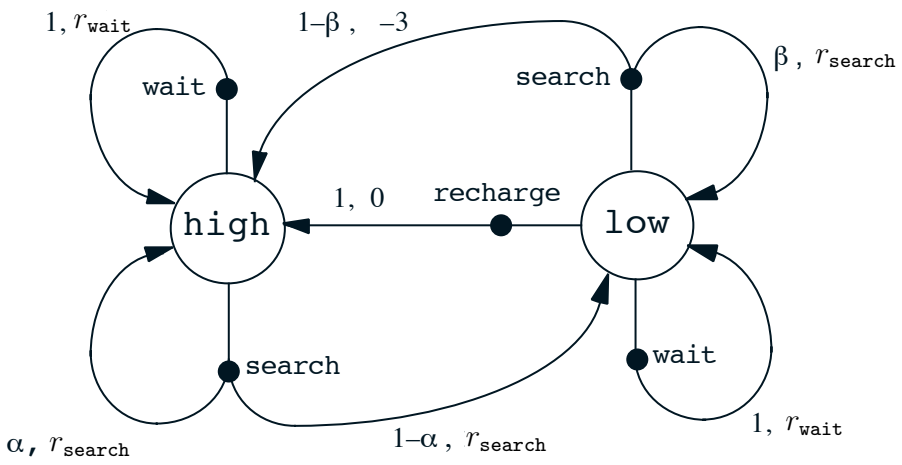
- dotado de batería, tracción, sensores para detección de latas, brazo articulado para coger las latas y cubo de recogida de latas
- puede buscar latas o esperar en reposo a que se depositen latas vacías en su cubo de recogida
- puede acudir a la estación de carga cuando la batería se está agotando
- simplificación: solo dos estados según la carga de la batería: *alta* (high, h) y baja (low, l) y tres acciones: *buscar* (search, s), *esperar* (wait, w) e *irArecargar* (recharge, c)
- cada lata recogida tiene una recompensa de 1, mientras que si agota la batería y ha de ser rescatado y cargado manualmente, la recompensa (castigo) es -3.
- una política óptima (a aprender por el sistema): buscar latas mientras la batería está alta y esperar pasivamente y/o ir a recargar cuando la batería está baja

Ejemplo: robot de reciclado: Probabilidades de transición y recompensas

q	q'	a	$\tau(q' \mid a, q)$	$r(q, a, q')$
high	high	search	α	r_{search}
high	low	search	$1-\alpha$	r_{search}
low	high	search	$1-\beta$	-3
low	low	search	β	r_{search}
high	high	wait	1	r_{wait}
high	low	wait	0	r_{wait}
low	high	wait	0	r_{wait}
low	low	wait	1	r_{wait}
low	high	recharge	1	0
low	low	recharge	0	0

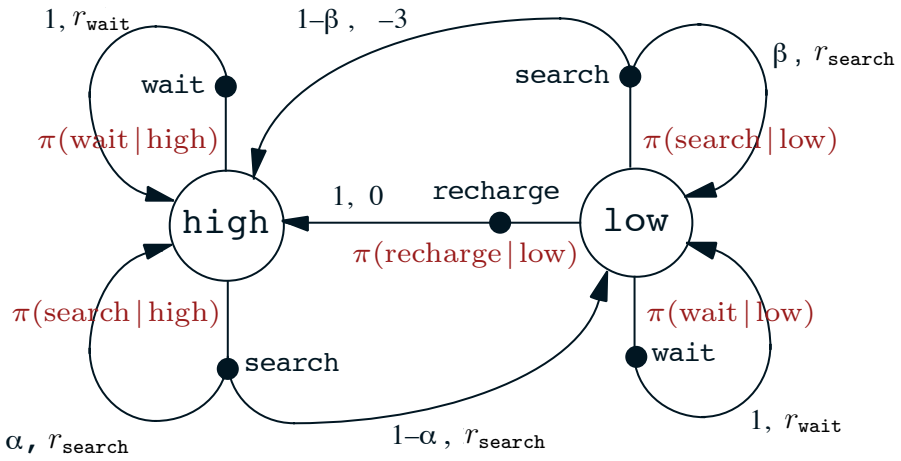
- r_{search}: número esperado de latas recogidas activamente
- r_{wait}: número esperado de latas recogidas pasívamente
- $r_{\text{search}} > r_{\text{wait}}$

Ejemplo: robot de reciclado: diagrama de transición



- r_{search}: número esperado de latas recogidas activamente
- r_{wait}: número esperado de latas recogidas pasívamente
- $r_{\text{search}} > r_{\text{wait}}$

Ejemplo: robot de reciclado: diagrama de transición



- r_{search}: número esperado de latas recogidas activamente
- r_{wait}: número esperado de latas recogidas pasívamente
- $r_{\text{search}} > r_{\text{wait}}$
- Política a aprender: $\pi(A \mid Q)$

Aprendizaje por refuerzo: políticas y valores

- Una *política* es una estrategia del agente para elegir las acciones en los sucesivos pasos de interacción. Se puede definir como una distribución de probabilidad $\pi(A_t = a \mid Q_t = q)$ (política estocástica) o como una función $(\pi : \mathbb{Q} \to \mathcal{A})$ (política determinista)
- El *valor de un estado* q para una política π es la esperanza matemática de la ganancia que se puede obtener ejercitando π a partir de q:

$$V_\pi(q) \;=\; \mathbb{E}_\pi[G_t \mid Q_t = q] \;=\; \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^t R_{t+k+1} \mid Q_t = q\right]$$

- Ecuaciones recursivas de Bellman

$$V_\pi(q) \;=\; \ldots \;=\; \sum_a \pi(a \mid q) \sum_{q'} \tau(q' \mid q, a)\,[r(q, a, q') + \gamma V_\pi(q')] \quad \forall q \in \mathcal{Q}$$

- Una *política óptima* $\pi^*(q)$ es aquella que conduce a un valor máximo de $V^*(q)$:

$$V^*(q) = \max_\pi V_\pi(q)\,; \qquad \pi^*(a \mid q) = \arg\max_\pi V_\pi(q)\,, \quad a \in A$$

Notación

- $\mathcal{A}$: algoritmo de aprendizaje
- $\mathcal{G}$: clase de funciones a aproximar o "aprender"
- $\mathcal{F}$: Clase funciones con las que se representan los "modelos"
- S: muestra finita de aprendizaje ($S \subset \mathcal{X} \times g(\mathcal{X})$)
- $|g|$: talla de la descripción de g
- $\ln$ y $\log$: logaritmo neperiano y decimal, respectivamente
- $P(x \mid \theta)$: probabilidad de $x \in \mathcal{X}$ dado por un modelo de parámetros dados θ
- $\max_x(.)$ ($\min_x(.)$) y $\arg\max_x(.)$ ($\arg\min_x(.)$): operadores que devuelven el máximo (mínimo) variando x y el argumento x que maximiza (minimiza) una función, respectivamente
- $\prod$ y $\sum$: operadores productorio y sumatorio, respectivamente
- $L_S(.)$: logaritmo de la verosimilitud de un conjunto S
- $\phi : \mathcal{X} \to \mathbb{R}$: una función discriminante
- $\mathcal{Q}$ conjunto de estados.
- r y R funciones de recompensa y recompensa acumulada, respectivamente
- τ: distribución de probabilidad de las transiciones