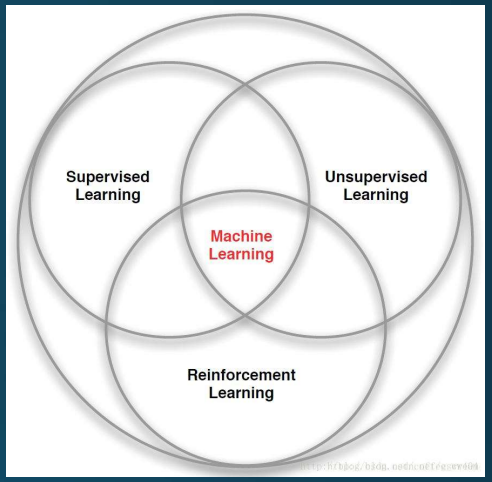
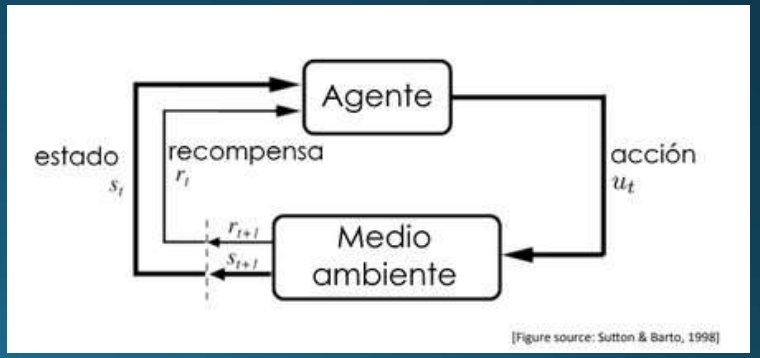
Josep Suy
Gener 2022

Aprenentatge reforçat

Introducció I.

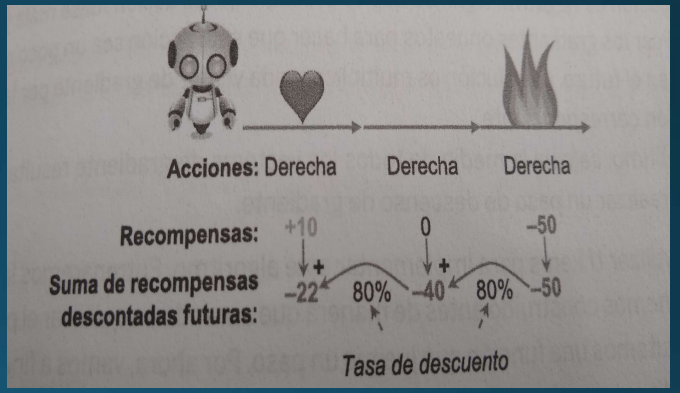
- Es basa amb la psicologia conductista. Condicionament operant.
- Reforçament positiu o negatiu. Recompensa.
- Maximitzat les recompenses.
- Prova i error prenen nota de les decisions per anar millorant. Feedback.
- Equilibri entre exploració i explotació.
- Tenim:
  - Agent (programa)
  - Entorn o medi ambient
  - Estat
  - Acció
  - Recompensa
- https://www.youtube.com/watch?v=QilHGSYbjDQ&t=5s
- Historia:
  - 1951 Marvin Minsky (navegar per un laberint)
  - 2013 Empresa DeepMind jocs Atari
  - 2016 AlphaGo
  - 2019 AlphaZero (Go, Escacs, Shogi)
- Utilitzarem:
  - Aurélien Géron. Hands-On Machine Learning with Scikit-Learn, Keras & Tensorflow. 2na Edició. O'REILLY. Anaya
  - https://github.com/ageron/handson-ml2. Capítol 18.





Introducció II.

- L'algoritme de l'agent per determinar les seves accions és diu "política"
- Que entrenem -> Paràmetres o dades de la política.
- Com ho fem:
  - Força bruta
  - Algorismes genètics.
  - Política de Gradients (Xarxes Neuronals)
  - Decisions de Markov. Q-Learning. Q-Learning profund (Xarxes Neuronals)
- Per entrenar s'ha de tenir un entorn (mon simulat o mon real)



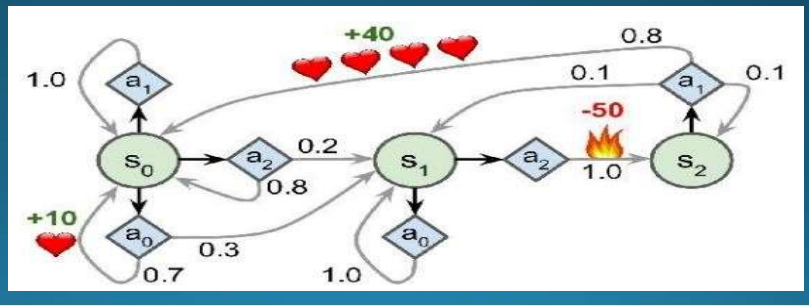
- Avaluar Accions. Assignació de crèdit. Quines accions han de rebre el crèdit (o la culpa) de la recompensa
  - Suma de las recompenses(retorn) futures descomptades : Quina responsabilitat (mèrit o culpa) té l'elecció d'una acció en el resultat final (veure imatge). Apliquem la taxa(factor) de descompte (1 a llarg plaç ,0 a curt plaç)
  - Avantatja de l'acció per cada estat.

Decisions de Markov. Algorisme Q-value

- Modelitzable amb estats, accions que podem realitzar a cada estat i probabilitat de transició a un altre estat per a cada acció.
- Basat amb els treballs de Markov (cadenes, sense recompensa) i Bellman (cadenes de decisions, amb recompensa).
- https://www.youtube.com/watch?v=B_lK-P68_Zc
- L'algorisme **Q-value (funció de valor d'un estat i d'una acció)** calcularem per a cada parella estat-acció (Q(s,a)) és la **suma de les recompenses (retorn) futures descomptades** en mitjana que l'agent pot esperar si ha escollit des de s fer l'acció a, però abans de veure el resultat d'aquesta acció, assumint que actua de manera òptima després de fer aquesta acció.
- Calcularem els Q òptims mitjançant el procés iteratiu de l'imatge (sempre convergeix):

  $Q_{k+1}(s_o, a) = \sum_{s_d} T(s_o, a, s_d)\,[R(s_o, a, s_d) + \gamma max_j Q_k(s_d, j)\,]$

  - T es la probabilitat de transició del estat s origen al s destí suposem que escollim a
  - R es la recompensa del mateix
  - γ es el factor de descompte
- Un cop tenim Q òptim per obtenir la política òptima tant sols cal escollir en cada estat s l'acció a amb Q(s,a) més gran.



Algorisme Q-Learning

- A priori només coneixem els estats i les accions que podem fer però no les probabilitats de anar a cada estat destí ni les recompenses.
- L'agent juga (aleatòriament) i millorant gradualment les estimacions de Q
- Un cop s'ha obtingut Q suficient bo igual escollim com a política òptima en cada estat s l'acció a amb Q(s,a) més gran.
- Calcularem els Q òptims mitjançant el procés (veure imatge), on:
  r és una mitjana mòbil de les recompenses
  α és lasa d'aprenentatge. Pot modificar-se al llarg del temps.
- Convergència lenta
- Es off-policy. La política que anem descobrim no l'utilitzem la que s'executa (aleatòria). És pot millorar escollint a cada pas amb una probabilitat el valor més alt i sinó els altres.
- No escala bé. (exemple pac-man). Tenim masses estats . Utilització de paràmetres (distàncies, direccions,..)

$$NewQ(s,a) = Q(s,a) + \alpha[R(s,a) + \gamma max\, Q'(s',a') - Q(s,a)]$$

New Q value for that state and that action

Current Q value

Learning Rate

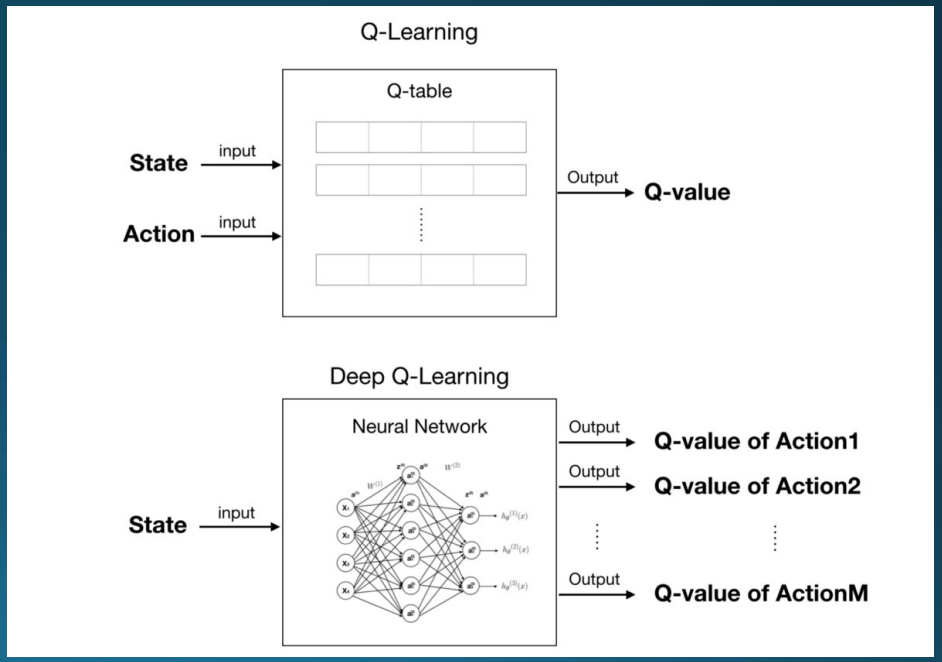
Reward for taking that action at that state

Discount rate

Maximum expected future reward **given the new s' and all possible actions at that new state**

Algorisme Q-Learning Profund. D.Q.N.

- La Q la substituirem per una xarxa neuronal que donada un estat ens doni per un estat s un valor (Q aproximat) per cada acció possible.
- Pot donar entrenament inestable (oblit catastròfic).
- Existeixen moltes variants:
  - Double DQN
  - Dueling DQN



Política de gradients

- A diferencia del Q que aprenem quin és la millor acció des de cada estat, els algorismes de política de gradients aprenen la funció de política pròpiament dita. Seguim els gradients per trobar els millors paràmetres de la política.
- Les accions poden ser deterministes (entorns deterministes, exemple escacs) o estocàstica (probabilitat).
- Avantatges:
  - Millor convergència, menys oscil·lació. Convergim a un màxim
  - Més efectiva per espais espais d'acció de gran dimensió o quan s'utilitzen accions contínues. Començarem a entendre quin serà el màxim, en lloc de calcular (estimar) el màxim directament a cada pas.
  - Poden aprendre polítiques estocàstiques. Produeix una distribució de probabilitat sobre les accions
- Desavantatges:
  - Poden convergir a un màxim local i no global.
- Algorisme general

    Inicialitzar paràmetres funció de política

    Per cada episodi fer:

        Calcular funció de cost

        Calcular gradients

        Actualitzar paràmetres segons gradient