An introduction to RL and deep RL

BABAK BADNAVA
MULTI-AGENT SYSTEMS LAB, IUST

مرکز تحقیقات هوش پارت

Different type of machine learning

- Supervised
  - Labeled data
- Unsupervised
  - Unlabeled data
- Semi-supervised
  - A small amount of labelled data with a large amount of unlabelled data
- Reinforcement learning
  - Data have no label, but there is a feedback.

Reinforcement learning

- Agent-oriented learning
- Learning by interacting with an environment to achieve a goal
- Learning by trial and error, with only delayed evaluative feedback (reward)
- The kind of machine learning most like natural learning
- Learning that can tell for itself when it is right or wrong

Reinforcement learning (2)

- An agent observes the environment
- Makes a decision
- And gets a feedback

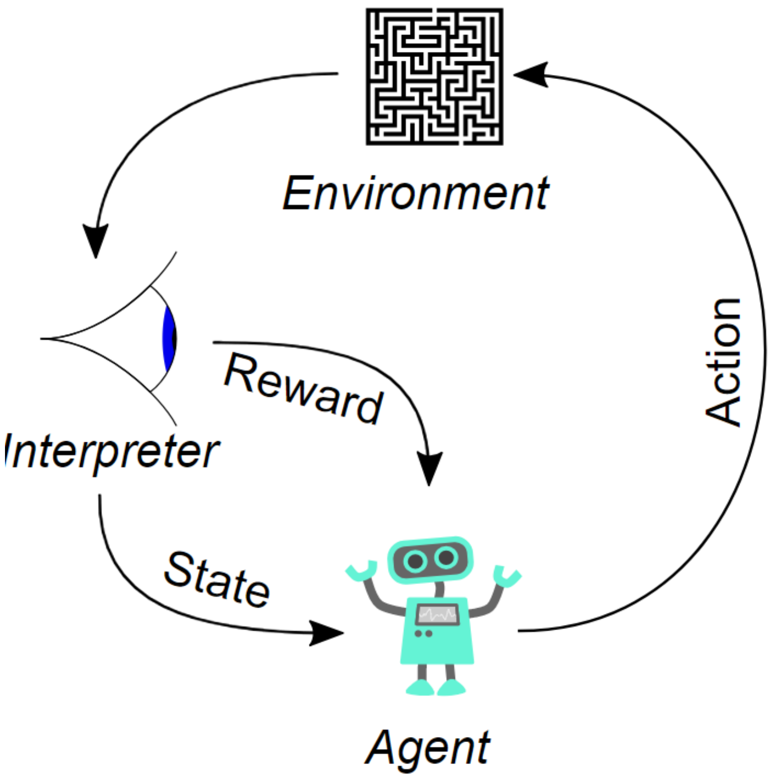
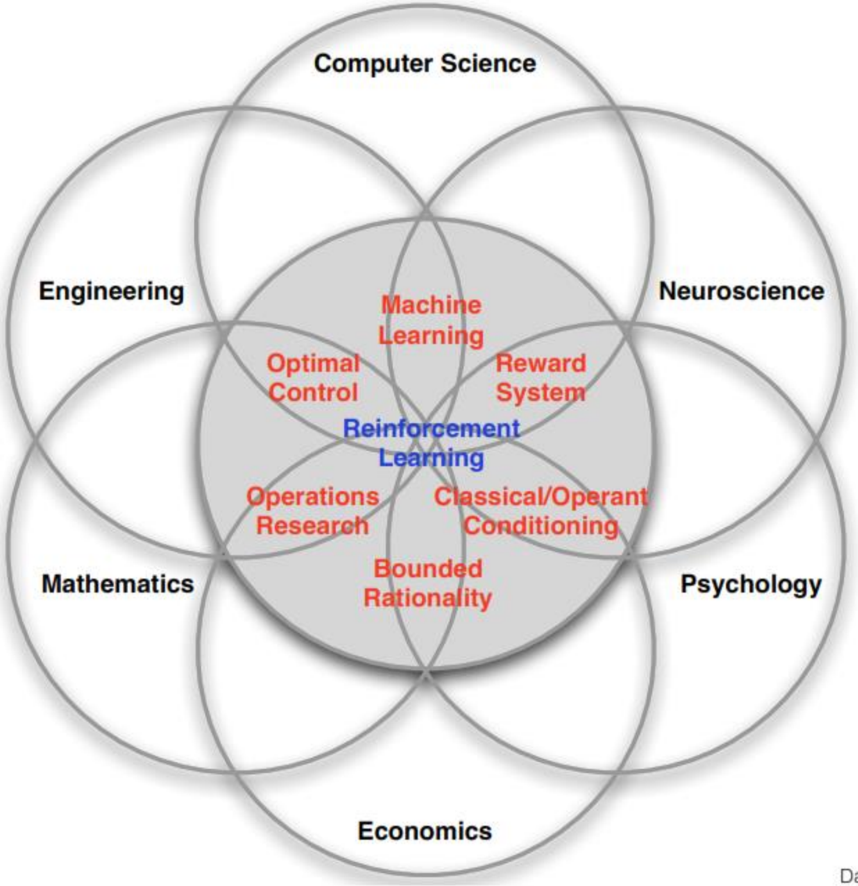


Image taken from Wikipedia.org

Where it came from?



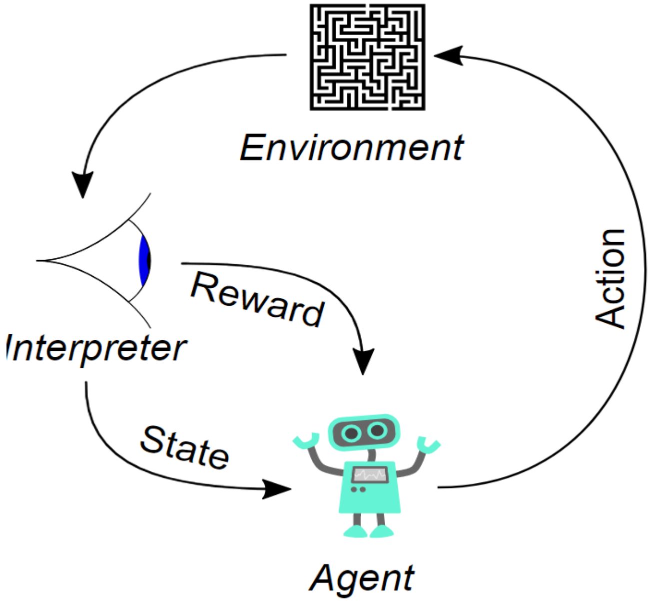
David Silver 2015

Applications

- Robotics
- Personalized web services
- Neuroscience
- Autonomous cars
- …

How to formulate?

- Use Markov decision process (MDP) to formulate.
- MDP is a tuple: $(S, A, P, R, \ \gamma)$
- S is a set of all possible states
- A is a set of all actions
- P is a transition function
- R is a reward function



Example: autonomous vehicle

- States: a set of all possible value for all sensors, speed, amount of available gas, …
- Actions: Pedals, Steering Wheels, …
- Reward: amount of money for an autonomous cab, distance to the destination, …..
- Transition function: natural rules + human-made rules
  - Gravity
  - Driving rules
  - …

Practice

- Formulate an MDP for an online shopping website who wants to increase its profit by giving its customers a discount.
- Formulate an MDP for a classification task like MNIST
- Formulate an MDP for a manufacturing robot

Policy function

- Map a state s to action $S \times A \rightarrow [0,1]$
- Can be deterministic or stochastic:
  - Deterministic: $\pi(s) = a$
  - Stochastic: $\pi(a|s) = P_{\pi}\,[A = a|S = s]$
- Example:
  - Car changing gear.

Value function

- How good is being in a state?
  - $V_\pi(s) = \mathbb{E}[G_t | S_t = s]$
  - $G_t = R_{t+1} + \gamma R_{t+2} + \ \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$
- Action-value function:
  - $Q_\pi(s,a) = \mathbb{E}[G_t | S_t = s, A_t = a]$
  - $V_\pi(s) = \sum_{a \in A} Q_\pi(s,a)\pi(a|s)$

Optimal value and policy

Optimal value functions:

- $V_*(s) = \max_{\pi} V_\pi(s)$
- $Q_*(s,a) = \max_{\pi} Q_\pi(s,a)$

- Optimal policy:
  - $\pi_* = \arg max_\pi V_\pi(\mathrm{s})$

مرکز تحقیقات هوش پارت

Different type of MDP and how to solve them?

- Model-based:
  - We know both the transition function and reward function.
  - Dynamic programming.
- Model-free:
  - No transition function.
  - No reward function.

Model-free

- Value estimation
  - Estimate the value function
  - Use a predefined soft policy function
    - Epsilon greedy
    - Boltzmann policy
- Policy approximation
  - Directly approximate the optimal policy function

مرکز تحقیقات هوش پارت

Value estimation

- Update value function till convergence
  - $V(s_t) \leftarrow (1-\alpha)V(s_t) + \alpha G_t$
  - $V(s_t) \leftarrow V(s_t) + \alpha(R_{t+1} + \gamma V(s_{t+1}) - V(s_t))$
- We can do this for action-value function too:
  - $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\big(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\big)$
  - We call this SARSA
  - This is an on-policy method
- Q-learning: an off-policy method:
  - $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\big(R_{t+1} + \gamma \, max_{a \in A} \, Q(s_{t+1}, a) - Q(s_t, a_t)\big)$

Soft policies

- Epsilon greedy:
  - $$\pi(a|s) = \begin{cases} \varepsilon \ if \ a = \arg max_a Q(s,a) \\ \frac{1-\varepsilon}{|A|-1} \end{cases}$$
- Boltzmann (softmax):
  - $$\pi(a|s) = \frac{e^{-Q(s,a)/\tau}}{\sum_{a \epsilon A} e^{-Q(s,a)/\tau}}$$

Let's put it all together: SARSA

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Algorithm from Sutton 2018

مرکز تحقیقات هوش پارت

Let’s put it all together: Q-learning

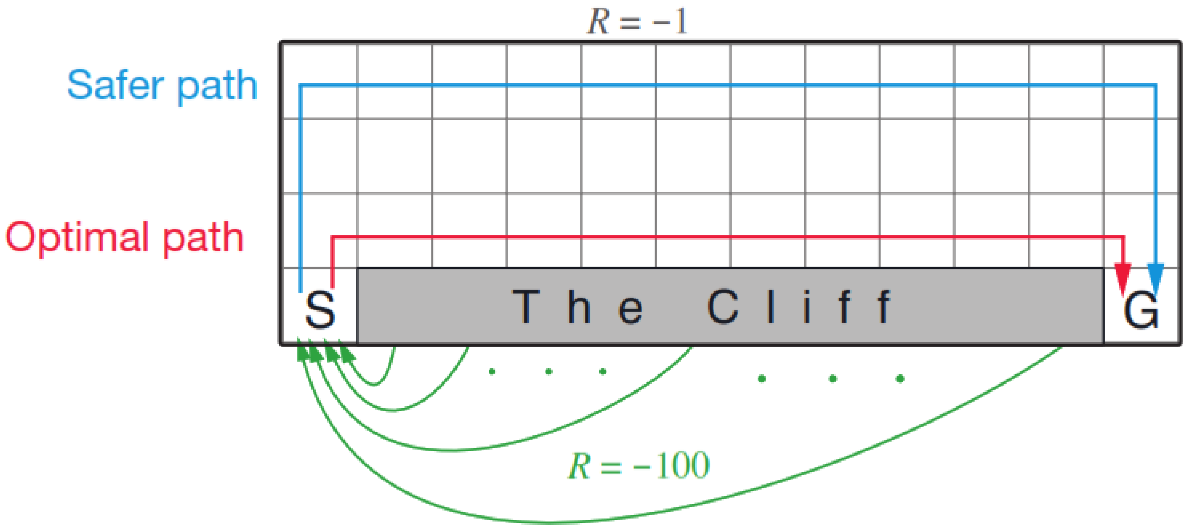
Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using policy derived from Q (e.g., ε-greedy)
        Take action A, observe R, S'
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
        $S \leftarrow S'$
    until S is terminal

Algorithm from Sutton 2018

A sample python code



Sutton 2018

Practice

- Try to implement SARSA and Q-learning for a continuous environment like Mountain Car.

References

- An introduction to Reinforcement Learning by Sutton 2nd edition