CS-C1000 – Introduction to Artificial Intelligence

Reinforcement Learning

Arno Solin

March 26, 2021

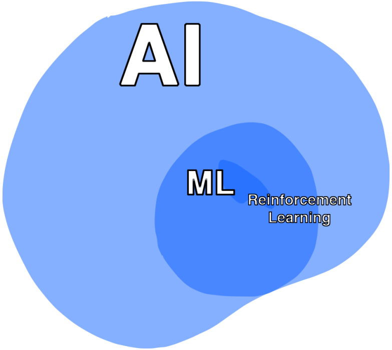
@arnosolin

arno.solin.fi

Outline and intended learning goals

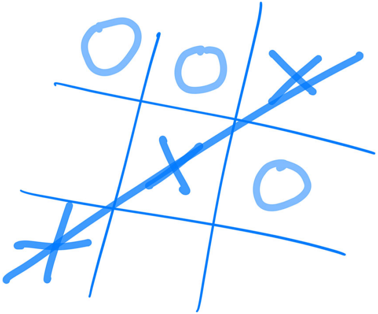
- What is Reinforcement Learning?
- Examples RL use cases
- Model-based methods
- Inverse Reinforcement Learning

AI → ML → Reinforcement Learning



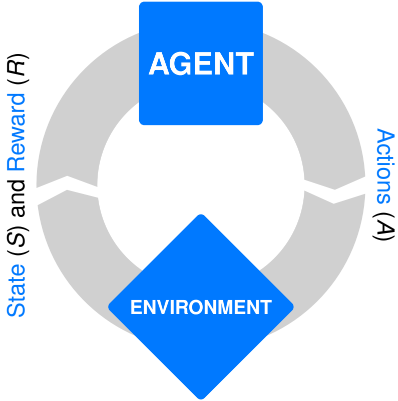
Reinforcement learning

- Concerned with how algorithms ought to take actions to maximize some cumulative reward.
- Typical uses cases where the task and environment are complicated, but some reward (and feedback) can be formulated.
- Example: Learning to walk, drive, grasp things, play games.



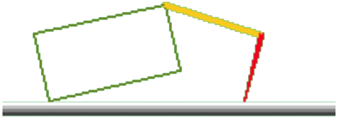
Basic idea

- Receive feedback in the form of rewards.
- Utility is defined by a reward function.
- Must (learn to) act so as to maximize expected rewards.
- All learning is based on observed samples of outcomes.



Concepts

- Actions:
  Button presses, muscle movement, etc.
- State:
  What the environment is like now.
  There might be stochasticity.
- Reward:
  A price or a penalty.



Reward $+1$ if moves to left
Reward -1 if moves to right

Figure: Dan Klein and Pieter Abbeel

Policy

- The agent's action selection is modeled as a map called policy:
$$\pi(a \mid s)$$
- The policy map gives the probability of taking action a when in state s.
- There are also non-probabilistic policies.
- The policy needs to be learned.

Discounting future rewards

- The agent tries to maximize the expected return of future rewards.
- The future rewards are discounted by the number of steps it takes to reach them.
- For example, with a discounting factor of $\gamma = 0.5$, the rewards one-step away are worth 1, and rewards two steps away only 0.5, for three 0.25, etc.

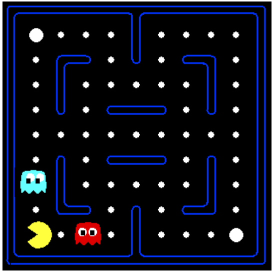


Figure: Dan Klein and Pieter Abbeel

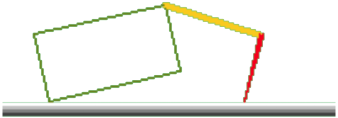
Exploration vs. exploitation

Explore further

Exploit what you already know

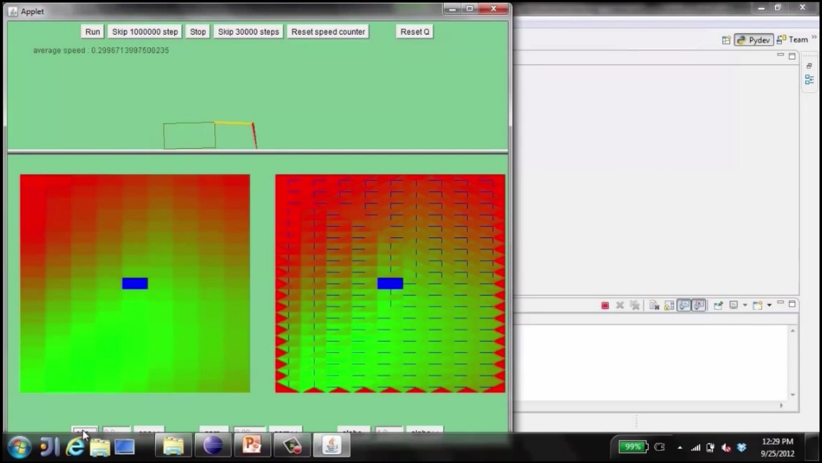
Concrete example: Q-learning

- Q-learning is a model-free reinforcement learning algorithm.
- You already tried this for the Tic-Tac-Toe game in the exercises.
- Let's see how it works on the 'crawler'.



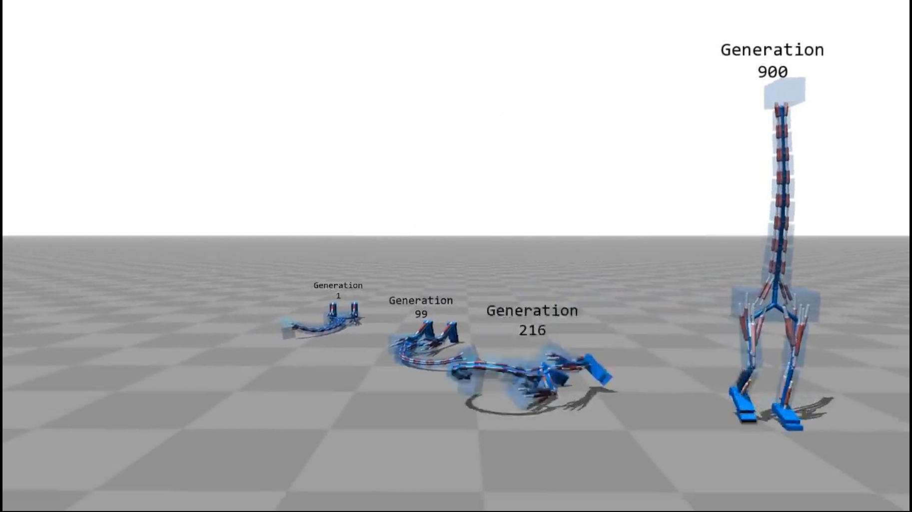
Reward $+1$ if moves to left
Reward -1 if moves to right

Figure: Dan Klein and Pieter Abbeel

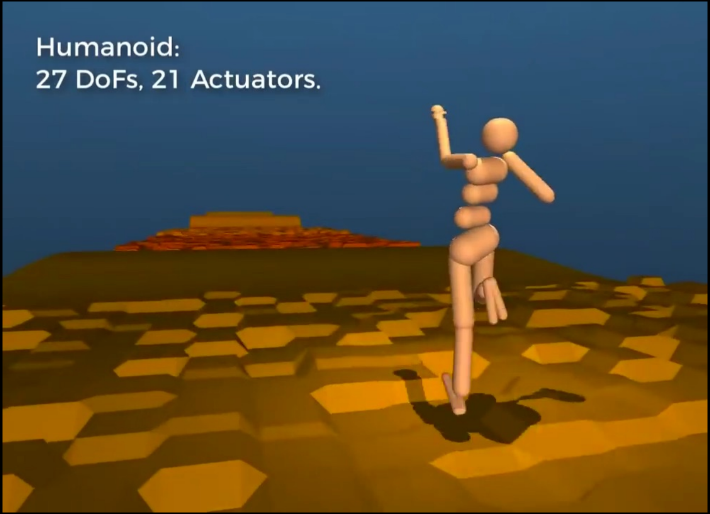


Video of Demo Q-learning Crawler :

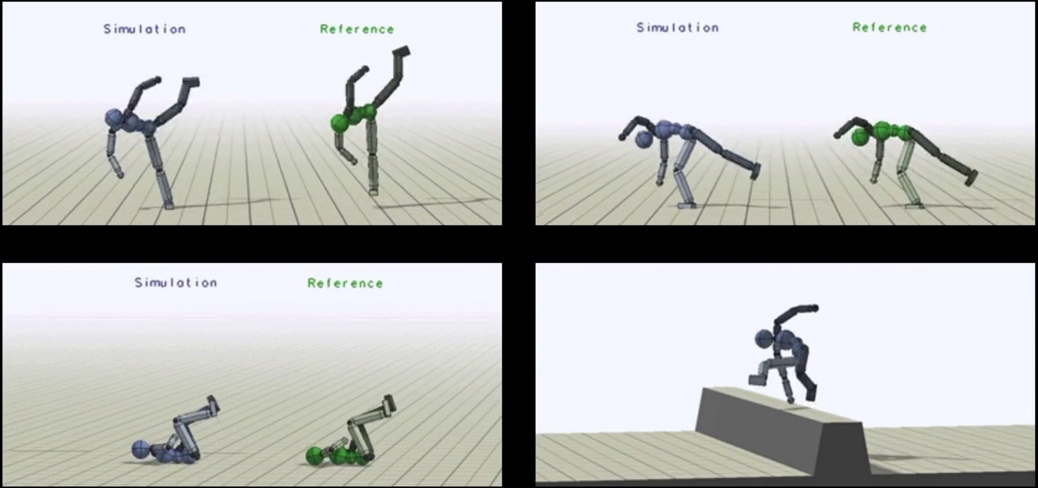
`https://www.youtube.com/watch?v=M2DQVWLxB7I`



Designing a control: Flexible Muscle-Based Locomotion for Bipedal Creatures:
`https://www.youtube.com/watch?v=pgaEE27nsQw`



Emergence of Locomotion Behaviours in Rich Environments:
https://www.youtube.com/watch?v=hx_bgoTF7bs



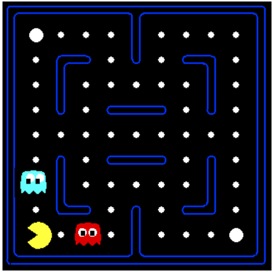
DeepMimic: Example-Guided Deep Reinforcement
Learning of Physics-Based Character Skills:
https://www.youtube.com/watch?v=vppFvq2quQ0

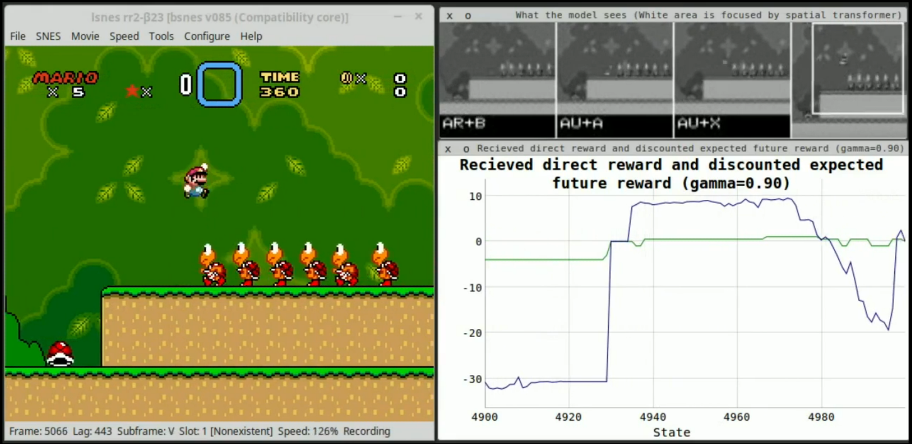
Deep Reinforcement Learning

- Traditionally, explicit design of state space and action space required, while the mapping from state space to action space is learned.

- Human designers have had to design how to construct state space from sensor signals and to give how the motion commands are generated for each action before learning.

- In Deep RL (or End-to-end RL), neural networks are used for learning everything, not just the actions, in an unsupervised way.

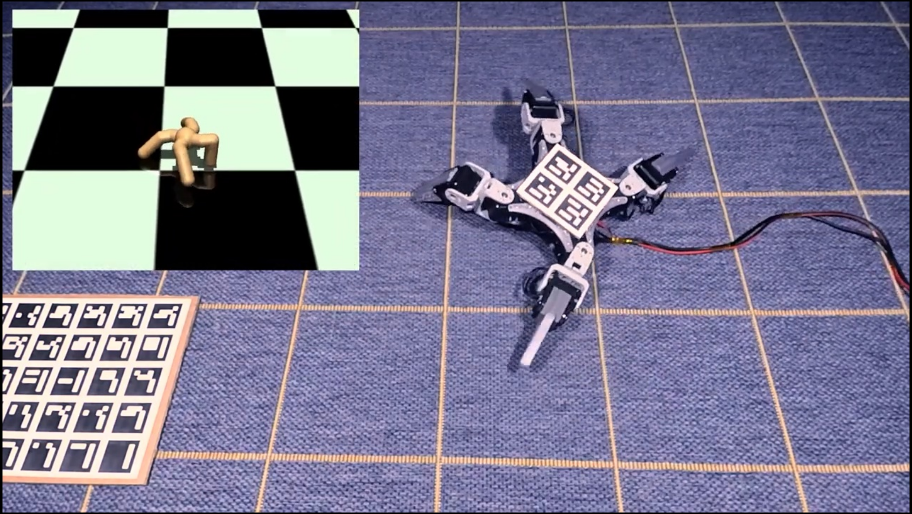




AI playing Super Mario World with Deep Reinforcement Learning:
https://www.youtube.com/watch?v=L4KBBAwF_bE

Model-based methods

- You might notice that most RL use cases nowadays are such that you can do the training in a simulator (playing millions of games, exploring in a physics simulator).
- Model-based RL provides additional knowledge:
  - Learn an approximate model based on experiences
  - Solve for values as if the learned model were correct
- This can be a lot faster than exploring for thousands and thousands of steps.



RealAnt: An Open-Source Low-Cost Quadruped for Research in Real-World Reinforcement Learning:
https://youtu.be/pG-XhH-9s7o

Inverse Reinforcement Learning

- In inverse reinforcement learning,
  no reward function is given.
- Instead, the reward function is inferred given an observed behavior from an agent.
- This can be interesting in user studies, UI development/optimization, etc.
  (finding out why ‘experts’ do what they do)

Recap

- Concerned with how algorithms ought to take actions to maximize some cumulative reward.
- Task and environment are complicated, but some reward (and feedback) can be formulated.
- Very active research topic.
- However, still few applications (that really work).

What next?

- Next week is different: The lecture is on Tuesday!
  - Guest lecturer Markus Ojala from Unity Technologies
  - + Normal lecture.
- Next computer exercise will be published on Tuesday
  but the next exercise session is on Tuesday in two weeks.

AI