Reinforcement Learning

Lecture 1: Foundations

Robert Lieck

Durham University

Lecture Overview

1. **Introduction**
   - definition
   - examples
   - comparison
2. **A Brief History**
   - learning by trial and error
   - optimal control and dynamic programming
   - monte carlo tree search
   - temporal difference algorithms
3. **Key Concepts**
   - designing rewards
   - action spaces
   - observability
   - information states
   - policies
   - value functions
   - model
   - taxonomy

Richard **Sutton** & Andrew **Barto** [5] summarise:

Definition: Reinforcement Learning

"Reinforcement learning is a computational approach to understanding and automating **goal-directed learning** and **decision making**. It is distinguished from other computational approaches by its emphasis on learning by an **agent** from direct **interaction with its environment**, without requiring exemplary supervision or complete models of the environment"

Reinforcement Learning

- Learn policies to
  - Play games and via self-play
- Learns optimal economic policies
  - AI economist
- Move from simulation to the real-world
  - Control robots e.g. Humanoids
- Surprising the creators!
  - Some examples

OpenAI with Gym

Introduction comparison to traditional machine learning

Typical Machine Learning

- Supervisory signal (with a teacher)
  - Immediate feedback
- Learning without a teacher
  - Unsupervised (e.g. clustering)
- i.i.d datasets

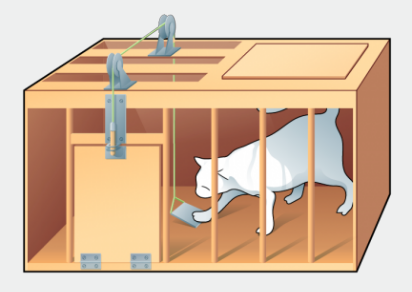
Reinforcement Learning

- Reward signal accumulated over time
  - Sparse/delayed feedback
- Not i.i.d
  - *sequential* where actions change subsequent environment

Prof. Barto gives an excellent history of the reinforcement learning field in this YouTube video

- Learning by **trial and ~~error~~ evaluation**
  - Edward L. Thorndike (1874-1949) Behaviourism. **Law of effect, 1911:** do something satisfying, then it becomes more probable. If its discomforting it becomes less probable.

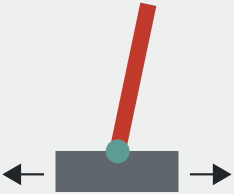
Thorndike's Puzzle Box



Also see *A Brief History of Intelligence* (Max Bennett, 2023) for a nice popular science book.

A Brief History optimal control and dynamic programming

Cart-Pole Balancing



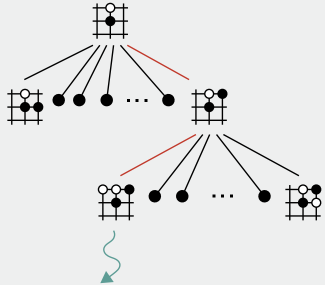
Barto et al., 1983: Neuronlike elements solve difficult learning control problems [1]

OpenAI with Gym

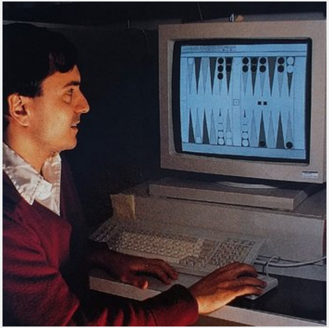
- Richard Bellman (1920-1984)
    - Optimal control theory
    - Dynamic programming, 1953
        - Breadth-first search through **state** space...
          how big is the state space of Go or StarCraft?
        - The Bellman Equation

- **Monte carlo** tree search
  - RL had a reputation of being slow
  - Gerald Tesauro showed in the 1990s multiple MC games can focus DP onto relevant parts of the state space

MCTS in AlphaGo Zero [4]



TD Gammon, 1992



Gerald Tesauro showed a **multi-layer neural network** with **TD learning** played competitively with human experts [6]

- **Temporal difference** learning
  - Connection to how dopamine cells work in neuroscience [2]
  - Monte Carlo require playing an entire game, TD methods adjust predictions to match later, more accurate, predictions about the future before finishing the game

Designing rewards is a key challenge in reinforcement learning

Definition: Reward

A **reward** $R_t \in \mathbb{R}$ is a *scalar feedback signal*

- How well the agent is doing at step t
- Agents try to maximize cumulative reward over time into the future

Definition: Reward hypothesis [5]

All goals and purposes can be thought of as the maximization of the expected value of the cumulative sum of a received scalar reward signal

Key Concepts action spaces

The RL challenge is to design an algorithm that chooses the action A_t given an observation O_t that maximizes (future) rewards.

Actions can be:

- **Discrete** – for example Go and chess
- **Continuous** – controlling voltage of a robot

RL Agents

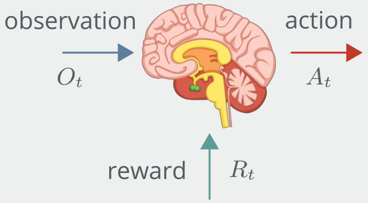
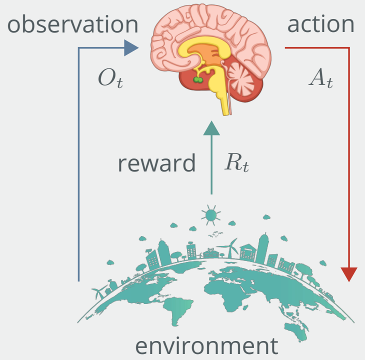


Figure based on [3, 5]

RL Agents



At step t, the agent:

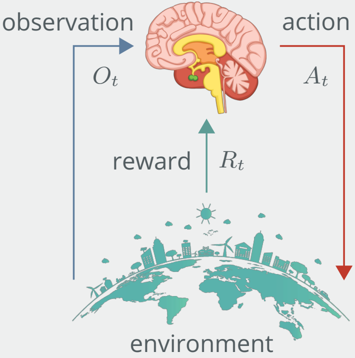
- Executes an **action** A_t

and also (without control):

- **Observes** O_t the environment
- Receives a **reward** R_t

Figure based on [3, 5]

RL Agents



The **environment** has a state S_t^e

- Typically not used
- Not all visible to the agent

The **agent** has a state S_t^a

- Summarises relevant observations
- Its any function of history $S_t^a = f(H_t)$

Definition: Full observability

This is where:

$$O_t = S_t^a = S_t^e,$$

unlike **partial observability** where $S_t^a \neq S_t^e$

Figure based on [3, 5]

Key Concepts information states

With the **Markov property** , we can throw away the history and just use the agents state:

Definition: Markov property

A state S_t is **Markov** if and only if

$$P(S_{t+1} \mid S_t) = P(S_{t+1} \mid S_1, S_2, ..., S_t)$$

- For example, a chess board
  - We don't need to know how the game was played up to this point
- The state fully characterises the distribution over future events:

$$H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty}$$

Agent component 1 :

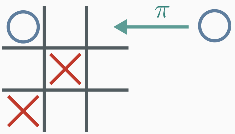
Definition: Policy

A **policy** is how the agent picks its actions. A policy π can be either **deterministic**, where:

$$a = \pi(s),$$

or it can be **stochastic**, where:

$$a \sim \pi(a|s).$$

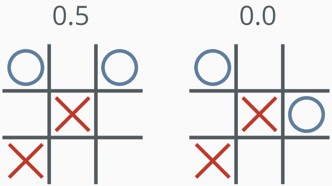


Agent component 2:

Definition: Value function

The **value function** is the prediction of expected total **future** rewards:

$$v_\pi(s) = \mathbb{E}_\pi[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots | S_t = s]$$



value for nought states

Agent component 3 :

Definition: Model

The **model** predicts what the environment will do next. It models the joint distribution of the new state and reward:

$$p(s', r|s, a) = P(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a).$$

The model is optional (**model-based** vs **model-free** learning)

Conclusion taxonomy

Taxonomy of reinforcement learning algorithms

- RL algorithms
  - Model-free
    - Policy optimisation
      - DPG
      - A3C
      - PPO
      - TD3
      - ...
    - Q-learning
      - DQN
      - Double DQN
      - Noisy DQN
      - TRPO
      - ...
  - Model-based
    - Learn model
      - World Models
      - I2A
      - MBMF
      - MBVE
      - ...
    - Given model
      - AlphaZero
      - ExIt
      - ...

This figure does not capture overlap, for example between policy optimsiation and Q-learning algorithms

References I

[1] A. G. Barto, R. S. Sutton, and C. W. Anderson.
Neuronlike adaptive elements that can solve difficult learning control problems.
IEEE transactions on systems, man, and cybernetics, SMC-13(5):834–846, 1983.

[2] W. Schultz, P. Dayan, and P. R. Montague.
A neural substrate of prediction and reward.
Science, 275(5306):1593–1599, 1997.

[3] D. Silver.
Reinforcement learning lectures.
https://www.davidsilver.uk/teaching/, 2015.

[4] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al.
Mastering the game of go without human knowledge.
nature, 550(7676):354–359, 2017.

References II

[5] R. S. Sutton and A. G. Barto.
Reinforcement learning: An introduction (second edition).
Available online, MIT press, 2018.

[6] G. Tesauro.
Temporal difference learning and TD-Gammon.
Communications of the ACM, 38(3):58–68, 1995.