Reinforcement Learning

Lecture 1 Intro & Overview

Nan Ye

School of Mathematics and Physics
The University of Queensland

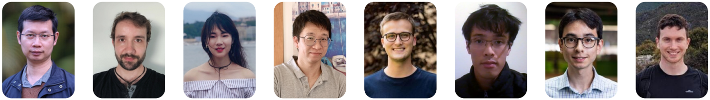
an academic...


teaching


proving


hacking


babysitting

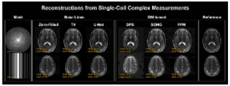

supervising some very intelligent people

cartoons generated by AI

autonomous driving

routing behavior analysis

fishery stock assessment

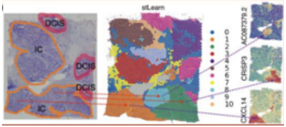


MRI reconstruction

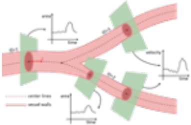
theoretically grounded practical algorithms
for
learning & decision-making

AAGI
Analytics for the Australian Grains Industry

agriculture analytics



spatial transcriptomics



haemodynamics modeling

These Lectures

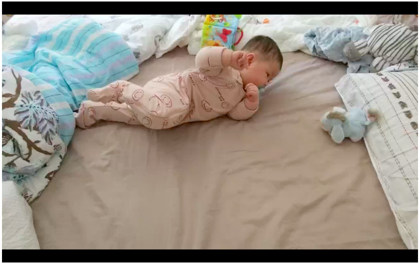
Reinforcement Learning (RL)

Goals

- cover mathematical & algorithmic foundation
- in-depth look at a few cool applications
- develop basic practical skills

The Journey Begins
from animal learning...

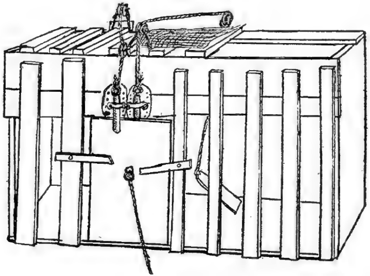
supervised learning
https://www.youtube.com/watch?v=F81VylqnzGE



reinforcement learning

learning to roll over

Edward Thorndike
Source: Wikipedia



Thornkdike's Puzzle Box
Source: Thorndike (1898, p. 8)

learning to escape

Thorndike, Animal intelligence: An experimental study of the associative processes in animals. 1898

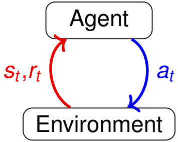
Thorndike's law of effect *(Thorndike, 1911, p. 244) Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond.*

in short: what works gets strengthened, what fails gets weakened.
or: trial and error learning / reinforcement learning (RL)

Thorndike, *Animal intelligence: Experimental studies*, 1911

to Artificial Intelligence (AI)...

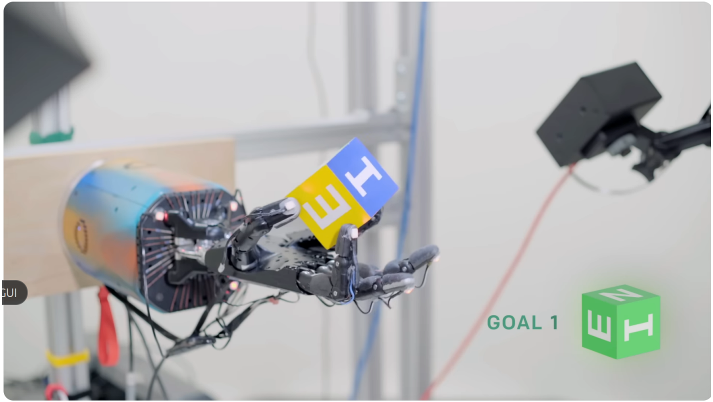
- Reinforcement learning (RL) in AI
  - many mathematical formulations of how an agent (algorithm) learns how to act in an unknown environment by interacting with the environment.
- At time t, the agent executes an action a_t, and the environment provides its state s_t and a reward r_t as the feedback.



- The goal is to learn a policy (mapping from state to action) that maximizes the expected rewards.

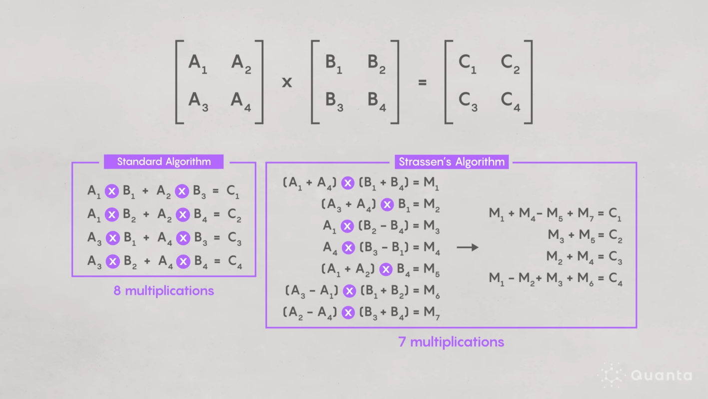
learning to play Atari games

https://www.youtube.com/watch?v=TmPfTpjtdgg



learning dexterity

https://www.youtube.com/watch?v=jwSbzNHGflM



learning fast matrix multiplication

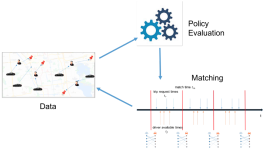
https://www.youtube.com/watch?v=fDAPJ7rvcUw


play Go (DeepMind)

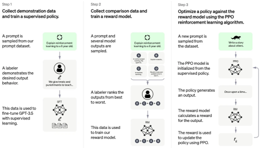

play StarCraft (DeepMind)


robot control (Boston Dynamics)


autonomous car (Waymo)



order dispatching (Qin et al., 2020)


ChatGPT (OpenAI)

many others: dialogue systems, healthcare, energy, . . .

Qin et al., Ride-hailing order dispatching at didi via reinforcement learning, 2020

Roadmap

- Introduction and overview

  motivation, bandits, big picture
- Classical ideas

  temporal difference methods, policy gradient, . . .
- Deep Reinforcement learning

  neural networks, DQN, DDPG, . . .
- Advanced techniques

  representation learning, stabilization, few-shot learning
- Applications

  AlphaGo, AlphaTensor, . . .

Devil Slayer

A PHD (Poetic Hero of Downunder) is tasked to slay a devil called Dilemma.

The PHD can attack using a sword or a shield, with a random damage.

The devil can only nullify an attack with a fixed probability.

The damage distributions are unknown. What should the PHD do to maximize the damage?

(a) Always use the sword.

(b) Always use the shield.

(c) 10x sword, 10x shield, then always the one with higher average.

(d) Throw a coin to decide for each attack.

(e) None of the above.

many other similar problems (Bouneffouf and Rish, 2019)

- clinical trials
- dynamic pricing
- recommender systems
- algorithm selection
- ...

these are formulated as multi-armed bandits

Multi-armed Bandits (MABs)

What's the best sequence of pulls for K bandits (slot machines)?

Source: Wikipedia

- Various formulations

  stochastic bandits, adversarial bandits, Markovian bandits, contextual bandits
- We focus on stochastic bandits satisfying the following assumptions
  - fixed but unknown reward distributions with means $\mu_1, \ldots, \mu_K$
  - for each pull, a reward is sampled from the pulled arm's distribution, independently from the past
  - bounded rewards in [0, 1]
- Best strategy: pull the arm with the highest mean reward

  $\Rightarrow$ not achievable as reward distributions and their means unknown

  $\Rightarrow$ need to explore (try less played arms) and exploit (play rewarding arms).

Regret Minimization

- We would like to play to minimize the expected regret

$$R_T = T\mu_* - \mathbb{E}[\sum_{t=1}^{T} r_t],$$

  where T is the number of pulls, $\mu_* = \max_i \mu_i$, and r_t is the reward at time step t.
- Alternatively, the expected regret is

$$R_T = T\mu_* - \sum_{k=1}^{K} \mu_k \, \mathbb{E}[n_k(T)],$$

  where $n_k(T)$ is the number of pulls for k at time step T.
- Lower bound: R_T is at least of the order $O(\ln T)$ (Lai and Robbins, 1985).

Lai and Robbins, Asymptotically efficient adaptive allocation rules, 1985

Upper Confidence Bound (UCB)

Algorithm UCB (Auer, Cesa-Bianchi, and Fischer, 2002)

1: **for** $t = 1, 2, \ldots$ **do**
2: play machine j with maximum

$$\bar{x}_j + \sqrt{\frac{2 \ln t}{n_j}},$$

where

$$\bar{x}_j = \text{average reward for machine } j,$$
$$n_j = \text{number of plays for machine } j.$$

- an example of optimism in the face of uncertainty
- each arm is played infinitely many times

Auer, Cesa-Bianchi, and Fischer, Finite-time analysis of the multiarmed bandit problem, 2002

(Auer, Cesa-Bianchi, and Fischer, 2002, Theorem 1) Given K machines with arbitrary reward distributions with support in $[0, 1]$, the expected regret of UCB is

$$R_T = \left[8 \sum_{k:\mu_k<\mu^*} \left(\frac{\ln T}{\Delta_k}\right)\right] + \left(1 + \frac{\pi^2}{3}\right)\left(\sum_{k=1}^{K} \Delta_k\right) \in O(\ln T),$$

where $\Delta_k = \mu_* - \mu_k$, and μ_k is the expected reward for machine k.

Since the expected regret is at least $O(\ln T)$, UCB is optimal.

ϵ_t-greedy

Algorithm ϵ_t-greedy (Auer, Cesa-Bianchi, and Fischer, 2002)

Require: $d \in (0, \min_{k:\mu_k<\mu^*} \Delta_k]$, any $c > 0$

1: **for** $t = 1, 2, \ldots$ **do**

2: play $j^* = \text{argmax}_j \, \bar{x}_j$ w.p. $1 - \epsilon_t$, and play a random arm w.p. ϵ_t, where

$$\epsilon_t = \min\left(1, \frac{cK}{d^2 t}\right).$$

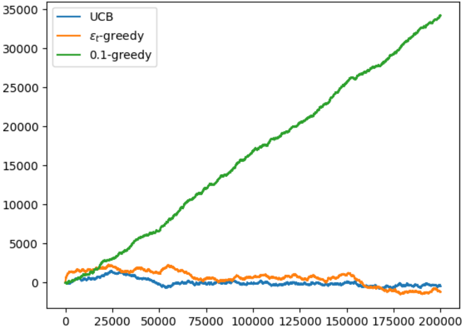
Does constant ϵ_t work? No!

(Auer, Cesa-Bianchi, and Fischer, 2002, adapted from Theorem 3) Given K machines with arbitrary reward distributions with support in $[0, 1]$, for large enough c, the expected regret of ϵ_t-greedy satisfies

$$R_T \leq \alpha \ln T,$$

for some $\alpha > 0$.

Original theorem (stronger): the probability of pulling a suboptimal arm is $O(1/t)$ at time step t.



$T\mu_* - \sum_{t=1}^{T} r_t$ against t on Devil Slayer in a simulation

Key Concepts

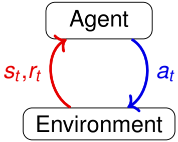
- exploration-exploitation tradeoff
- optimism in the face of uncertainty
- ϵ-greedy

these are important for RL in general

- UCT (Kocsis, Szepesvári, and Willemson, 2006) and POMCP (Silver and Veness, 2010) are UCB's extensions to MDPs and POMDPs
- (later) ϵ-greedy is commonly used in RL for MDPs

Reinforcement Learning

- Recall: in RL, an agent (algorithm) learns how to act in an unknown environment by interacting with the environment.



  Bandits are stateless.

- General structure of RL algorithms:

  RL = loop(experience collection + incremental learning)

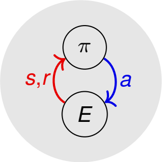
  1: **repeat**
  2:     collect experience
  3:     incremental learning
  4: **until** termination condition is met

environment model

bandits, MDPs, POMDPs

learning target

model, value, policy



behavior policy

exploration vs exploitation

update rules

experience, loss

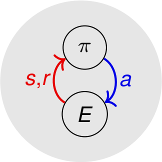
four dimensions

we will focus on RL for MDPs

policy evaluation / prediction

$\pi, E/\text{interactions} \;\rightarrow\; V_\pi$

value iteration, linear system, Monte Carlo, …



planning / control

$E \;\rightarrow\; \text{argmax}_\pi\, V_\pi$

value iteration, policy iteration, Monte Carlo, …

reinforcement learning

interactions with $E \rightarrow \text{argmax}_\pi\, V_\pi$

Q-learning, SARSA, policy gradient, …

π = *policy*, V_π = *policy value*, E = *environment*

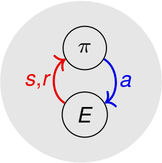
three interconnected problems

RL algorithms often rely on techniques for evaluation and planning

policy evaluation / prediction

$\pi, E/\text{interactions} \;\rightarrow\; V_\pi$

value iteration, linear system, Monte Carlo, ...



planning / control

$E \;\rightarrow\; \text{argmax}_\pi\, V_\pi$

value iteration, policy iteration, Monte Carlo, ...

reinforcement learning

interactions with $E \rightarrow \text{argmax}_\pi\, V_\pi$

Q-learning, SARSA, policy gradient, ...

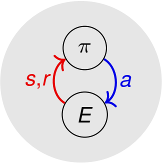
π = *policy*, V_π = *policy value*, E = *environment*

eval $\rightarrow$ plan: policy iteration (evaluate a policy, improve greedily)

policy evaluation / prediction

$\pi, E/\text{interactions} \rightarrow V_\pi$

value iteration, linear system, Monte Carlo, ...



planning / control

$E \rightarrow \operatorname{argmax}_\pi V_\pi$

value iteration, policy iteration, Monte Carlo, ...

reinforcement learning

interactions with $E \rightarrow \operatorname{argmax}_\pi V_\pi$

Q-learning, SARSA, policy gradient, ...

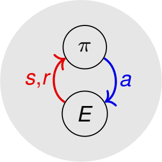
π = *policy*, V_π = *policy value*, E = *environment*

eval $\rightarrow$ RL: evaluate a policy using samples, improve policy

policy evaluation / prediction

$\pi, E/\text{interactions} \;\rightarrow\; V_\pi$

value iteration, linear system, Monte Carlo, ...



planning / control

$E \;\rightarrow\; \text{argmax}_\pi\, V_\pi$

value iteration, policy iteration, Monte Carlo, ...

reinforcement learning

interactions with $E \rightarrow \text{argmax}_\pi\, V_\pi$

Q-learning, SARSA, policy gradient, ...

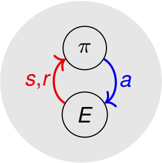
π = *policy*, V_π = *policy value*, E = *environment*

plan → RL: model-based RL (learn a model, then plan)

policy evaluation / prediction

$\pi, E/\text{interactions} \;\rightarrow\; V_\pi$

value iteration, linear system, Monte Carlo, ...



planning / control

$E \;\rightarrow\; \text{argmax}_\pi\, V_\pi$

value iteration, policy iteration, Monte Carlo, ...

reinforcement learning

interactions with $E \rightarrow \text{argmax}_\pi\, V_\pi$

Q-learning, SARSA, policy gradient, ...

π = policy, V_π = policy value, E = environment

RL $\rightarrow$ plan: run RL using an environment simulator

Roadmap

- Introduction and overview

  motivation, bandits, big picture

- Classical ideas

  temporal difference methods, policy gradient, ...

- Deep Reinforcement learning

  neural networks, DQN, DDPG, ...

- Advanced techniques

  representation learning, stabilization, few-shot learning

- Applications

  AlphaGo, AlphaTensor, ...

References I

- Auer, P., N. Cesa-Bianchi, and P. Fischer (2002). Finite-time analysis of the multiarmed bandit problem. In: *Machine learning* 47.2, pp. 235–256.
- Bouneffouf, D. and I. Rish (2019). A survey on practical applications of multi-armed and contextual bandits. In: *arXiv preprint arXiv:1904.10040*.
- Kocsis, L., C. Szepesvári, and J. Willemson (2006). Improved Monte-Carlo Search. In: *Univ. Tartu, Estonia, Tech. Rep* 1.
- Lai, T. L. and H. Robbins (1985). Asymptotically efficient adaptive allocation rules. In: *Advances in applied mathematics* 6.1, pp. 4–22.
- Qin, Z. et al. (2020). Ride-hailing order dispatching at didi via reinforcement learning. In: *INFORMS Journal on Applied Analytics* 50.5, pp. 272–286.
- Silver, D. and J. Veness (2010). Monte-Carlo planning in large POMDPs. In: *Advances in Neural Information Processing Systems* 23, pp. 2164–2172.
- Thorndike, E. (1911). *Animal intelligence: Experimental studies*. The Macmilllan Company.

References II

Thorndike, E. L. (1898). Animal intelligence: An experimental study of the associative processes in animals. In: *The Psychological Review: Monograph Supplements* 2.4, p. i.