Can Machines Learn from Experience?

An Intro to Reinforcement Learning

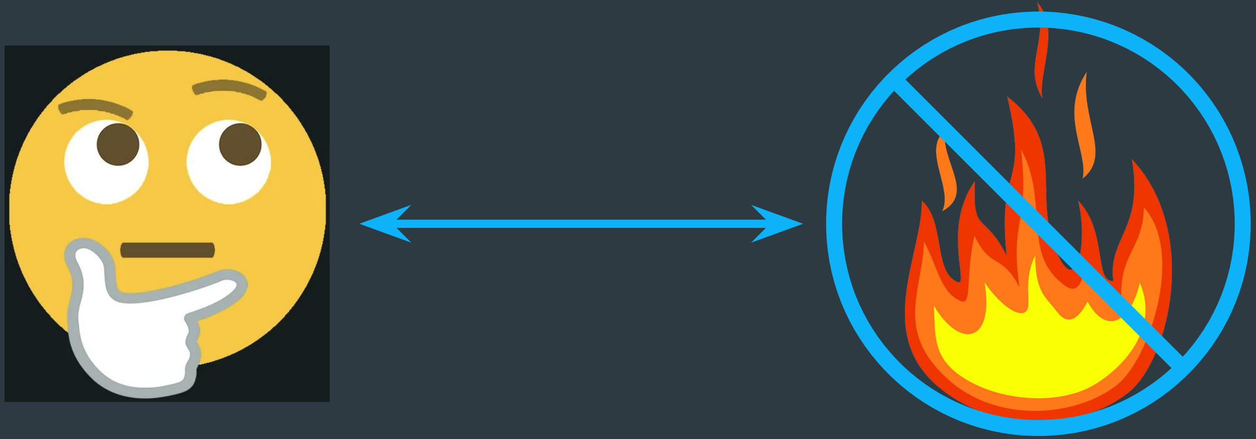
Agenda

1. Coordinator Pitches
2. The RL Problem (Ch. 1[1])
3. Markov Decision Processes (Ch. 3[1])

Agenda

1. Coordinator Pitches
2. The RL Problem (Ch. 1[1])
    a. Learning through interaction
    b. Reinforcement learning
3. Markov Decision Processes (Ch. 3[1])

Learning through interaction

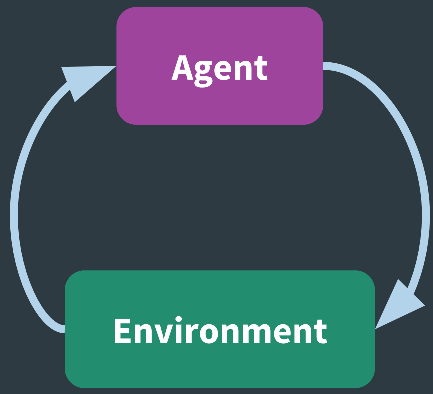


Reinforcement learning

- Formalization of learning through *interaction*
- What to do in order to maximize some numerical reward
  - Mapping states to actions, $\pi : S \rightarrow A$
- Characterized by the **problem** rather than a method(s)
  - Any method suited for RL problems can be considered an RL method

Reinforcement learning

- There is an agent-environment relationship
  - **A/E Interface**
- The agent
  - must be able to sense state
  - must be able to act to change that state
  - must have goal(s) related to that environment
- tuple: `(Sensation, Action, Goal)`



Elements of reinforcement learning

Policy	π
Reward Function	R
Value Function	Q
Model	T

π: maps from perceived states in the environment to actions that should be taken

R: defines the goal in a given problem, what the agent seeks to maximize – short-term focus

Q: also seeks to maximize, but has a long-term focus

T: mimics behavior of the environment; ultimately should allow for inference about how the environment will behave

A new paradigm

- We've covered ANNs, CNNs, and RNNs
  - These are typically used in supervised learning
- Supervised learning has an expert providing examples and labels
  - No learning through interaction
- Do we learn only from an expert's examples and those labels?
  - Handling novel situations, uncharted territory, etc.
    - Learning from experience is necessary
- **Key idea:** feedback is provided by *evaluating the actions taken*, rather than *instruction by correct actions*

Agenda

1. Coordinator Pitches
2. The RL Problem (Ch. 1[1])
3. Markov Decision Processes (Ch. 3[1])
    a. The Agent-Environment Interface
    b. Goals and Rewards
    c. The Markov Property
    d. Markov Decision Processes

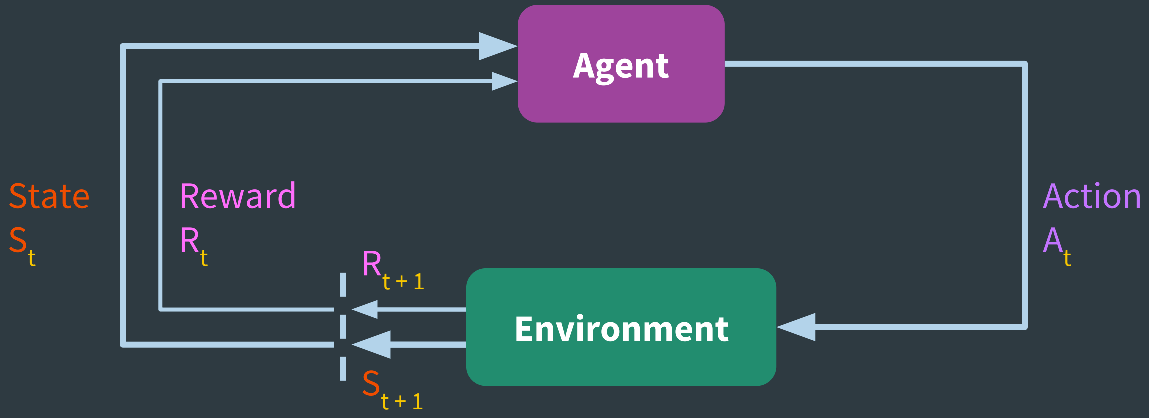
The Agent-Environment Interface

- Agent
  - is learner and decision maker

- Environment
  - intuitively: everything in agent *interacts with*
  - unintuitively: everything the agent is ***not***
  - gives rise to rewards (**R**), and has states (**S**)

- Task
  - single instance of the RL problem
  - complete definition of the environment and how rewards are determined

The A/E interface



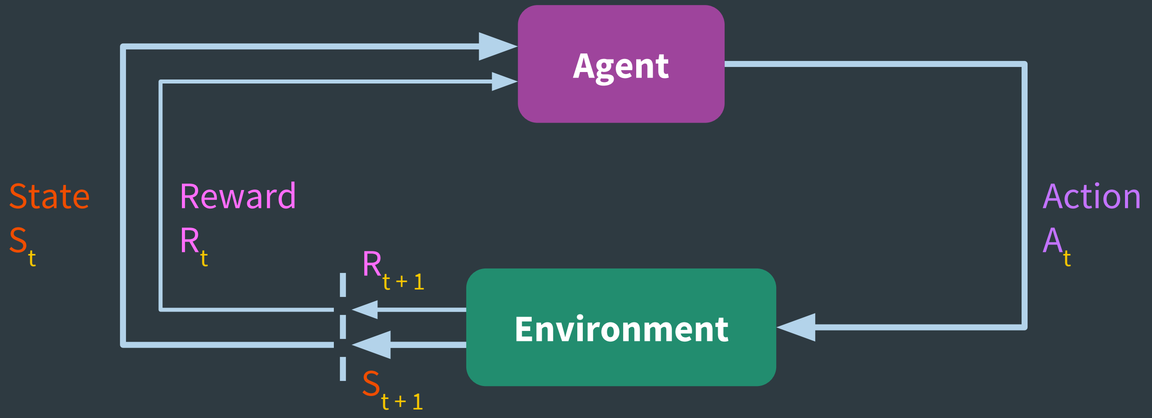
$t = 0, 1, 2, 3, \ldots$

$S_t \in \mathbf{S}$

$A_t \in \mathbf{A}(S_t)$

$R_t \in \mathbf{R}$

The A/E interface



$\pi_t : S_t \rightarrow A_t$

$\pi_t(a \mid s)$ is $P(A_t = a$ if $S_t = s)$

The A/E interface

- We deal with discrete timesteps
  - $t = 0, 1, \ldots, n \mid n \in \mathbb{Z}^+$
- At each time t the environment has some state (S_t)
  - $S_t \in \mathbf{S}$, where S is all possible states of the environment
- At each time t, given a state S_t, the agent takes an action (A_t)
  - $A_t \in \mathbf{A}(S_t)$, where $\mathbf{A}(S_t)$ is the set of actions available in S_t
- After taking an action A_t – the agent receives some reward in $t + 1$
  - $R_{t+1} \in \mathbf{R}$
- The goal of RL is to have an agent maximize reward over the long term

The A/E interface

- Abstract, flexible framework
  - Timesteps can refer to stages in decision making
  - Actions can be low- or high-level
  - States can be low- or high-level
- Choice in representation is “more art than science” – at least for now

Goals and Rewards

The Reward Hypothesis:

> *That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).*

Basically, we can define a goal as maximizing some numerical value

An example of goals and rewards

- Captures a lot of wanted behavior in situations/tasks
- Robot learning to walk
  - R_t = +n | n is proportional to forward motion at t
  - R_t = -1 for each timestep
  - max(R)
- A way to bias agents into *how* we want them to achieve their goals

Goals and Rewards

- Returns or Utility
- We’ve been informal so far
    - goals, maximizing cumulative reward
    - so what does that look like formally?

Goals and Rewards, Formally

- Returns
  - We seek to maximize *Expected Return/Utility*
- G_t is the function that defining some reward sequence
  - Simplest case: $G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$ | T is final timestep★

★ Having T makes sense for tasks that have are naturally finite
  - *think games*
  - each pass through the game to T is called an *episode* or *trial*

Aside: Episodic Tasks

- Episodic tasks are finite
  - they have an end
  - consider a game

- Continual tasks are infinite
  - no intuitive/conceivable/real end
  - consider learning about RL

Back to formalizing G_t

- Most tasks are continuous, we need a richer G_t
  - For most interesting tasks, $T = \infty$
  - Currently, G_t has an infinite value - not good - for continual tasks

- We need a terminal state
  - *Discounting*
  - Absorbing State*
  - Finite Horizon*

Aside: Discounting

- The notion that rewards now are better than rewards later
- $0 \leq \gamma \leq 1$, the *discount factor* (how much do future rewards matter?)
- $G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$ becomes: $\Sigma_{k=\{0..\infty\}} (\gamma^k R_{t+k+1}) \mid k \to \infty$
- Now, if $\gamma < 1$, G_t has a finite value!

  (provided $\{R_k\}$ is bounded)

Aside: Discounting

- Utility is $\Sigma_{k=\{0..\infty\}}(\gamma^k R_{t+k+1}) \mid k \rightarrow \infty$

- If $\gamma = 0$, the agent becomes myopic
  - doesn’t consider future rewards
  - $G_t = 0$ for $k > 0$

- As $\gamma \rightarrow 1$, the agent becomes increasingly farsighted
  - heavily weights future rewards
  - $G_t > 0$ for $k > 0$

Our return function so far

$$\Sigma_{k = \{0..\infty\}} (\gamma^k \cdot R_{t+k+1}) \mid k \rightarrow \infty$$

The Markov Property

- Agent makes decisions as a function of the state
  - state is a signal from the Environment (Env)
  - can be thought of as: info from the Env, available to the Agent
- What should be required of a State?
  - Give all relevant information
    - immediate sensations, and
    - past sensations
      (but not all of them)

The Markov Property

- A state that successfully retains all relevant info has the Markov Property
- The future is independent of the past, given the present, e.g.
  - cannonball’s trajectory mid flight, or
  - state of a checkers board mid game

So why's the Markov Property important?

- Consider: How might an Env respond at $t + 1$ for an action at t
- Previously: What will the state S_{t+1} be if I take some action in state S_t?
  $p(s', r \mid s, a) = \Pr\{S_{t+1} = s', R_{t+1} = r \mid S_0, A_0, R_0, \ldots, S_{t-1}, A_{t-1}, R_t, S_t, A_t\}$
- But, with the Markov Property,
  $p(s', r \mid s, a) = \Pr\{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\}$

The Markov Property

- If it is the case that, $p(s', r \mid s, a) = \Pr\{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\}$; then our task has the Markov Property
- This enables us to predict S_{t+1} and R_{t+1} given $S_t = s$ and $A_t = a$
- Can be shown just as powerful as having complete history
- Can be shown best policy for Markov is equal to best policy for complete histories

The importance of the Markov Property

- Assumed that decisions and values are functions of the current state
    - Thus, the state representation must be informative
- Theory helps understand behavior of the algorithms
    - Understanding of the theory of the Markov case is essential foundation
    - Assumption of Markov state not unique to reinforcement learning

Markov Decision Processes (MDPs)

- If a task possesses the Markov Property, then it's also a Markov Decision Process (MDP)

- If state and action space are finite, we call them a *finite MDP*
  - "... they are all you need to understand 90% of modern reinforcement learning."[1]

- Given one-step dynamics, $p(s', r \mid s, a)$, we can get:
  - Expected rewards for state-action pairs
  - State-transition probabilities
  - Expected reward of state-action-next-state triples

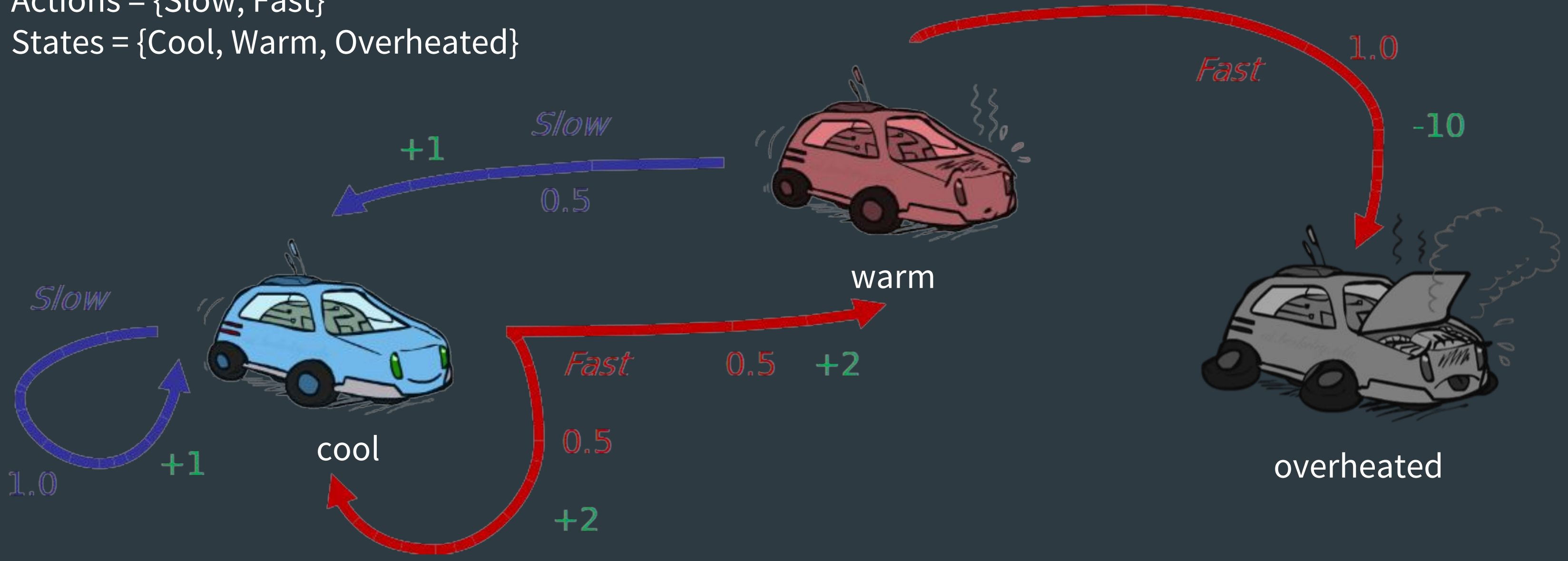
Markov Decision Processes

- A five-tuple (S, S_0, A, T, R)
  - $T = p(s' \mid s,a)$ or $T(s, a, s')$
  - $R(s,a,s')$ with discount factor

- A solution to an MDP is a policy $\pi: S \rightarrow A$
  - $V^*(s)$ - value of acting optimally from state s
  - $Q^*(s,a)$ - value of taking a in s and acting optimally thereafter
  - π^* - optimal policy

Markov Decision Process

Actions = {Slow, Fast}
States = {Cool, Warm, Overheated}



Graphics from CS188 at Berkeley

Markov Decision Processes

- Optimal Value Function $v_*(s)$
  - Satisfies a particular recursive relationship
  - $v_*(s) = \max_a v_*(s)$

- Optimal Action-Value Function Q(s)
  - $Q^*(s) = \Sigma_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$
  - $V^*(s) = \max_a \Sigma_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$

Bellman Optimality Equations

Markov Decision Processes

- $Q^*(s) = \Sigma_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$
- $V^*(s) = \max_a \Sigma_{s'} T(s, a, s')[R(s, a, s') + \gamma V^*(s')]$
- We want to iterate over an MDP to find V* and Q* and then find some π*

Questions?

References (Check ’em Out)

1. “Reinforcement Learning: An Introduction” - Sutton and Barto
2. CS188 at Berkeley

Resources (Check 'em Out)

- This Week in Machine Learning and AI (TWIMLAI)
- Mapping Babel
- Two Minute Papers
- Talking Machines
- CS188 at Berkeley by P. Abbeel
- Machine Learning on Coursera by A. Ng
- CS231 at Stanford by Fei-Fei Lei & Andrej Karpathy

On the Interwebs

goo.gl/Pq3tP4

<pending>

<pending>

goo.gl/FPuxtt