Welcome to Reinforcement Learning – Sp 24!
Lecture Outline

Course logistics and scope

What is RL, a formal definition

Why should we care?

Going beyond RL
Course Logistics

- **Where:** BAG 261
- **When:** Tue/Thu 11:30-12:50
- **Who:**
  - Abhishek Gupta (Instructor)
  - Patrick Yin (TA)
  - Zoey Chen (TA)
- **Office hours:**
  - Abhishek: Gates 215, Wed 2-3pm
  - Patrick:
  - Zoey:
Course Logistics

- Grading: Seminar style
  - 40% final project
  - 10% paper discussions/readings
  - 45% for HWs – 15% for each of 3 HWs
  - 5% participation

- Communications through EdStem/e-mail

- Mix of lectures and paper readings:

- Final projects will be presented in a poster session.
  - Intermediate project proposals and milestone check ins.

- Please participate, otherwise it will be boring for all of us!
Course Logistics - Project

- Final project (40% of grade):
  - Project proposal (1 page) [Due 4/10]
  - Milestone report (3-4 pages) [Due 5/2 (subject to change)]
  - Final report (6-8 pages) [Due 6/5 (subject to change)]

- Project can be investigating any question related to reinforcement learning, imitation learning or sequential decision making
  - New algorithm
  - Performant/stable implementation
  - Empirical investigation
  - New application or domain
  - …

- Can be done in groups of 1-2 students.
Course Logistics – Discussions

- We will try out a new discussion style (many to many role playing)
  - Stolen from Alec Jacobson (Toronto), Colin Raffel (UNC), etc

- Idea is that for every paper, we will have people fulfilling many roles:
  - **Discussion leader**: Presents the overall premise of the paper, quickly goes over the key ideas in the paper and presents the basic results
  - **Paper reviewer 1 (pro)**: Argues why the paper should be accepted
  - **Paper reviewer 2 (con)**: Argues why the paper should NOT be accepted
  - **Archaeologist**: Identifies where this paper fits into the literature
  - **Academic researcher**: Identifies future projects that can build on this work
  - **Industry practitioner**: Identifies where exactly this work can see application and discuss the societal impact of such an application
  - **Hacker**: Implements the algorithm/tries out the code if published

- Everyone else posts commentaries about the paper on EdStem!

- For non discussion lectures, everyone is expected to post a commentary on at least one of the readings.
Course Logistics – Homework

- 3 HW assignments, each Python programming of different algorithms
  - HW 1 – Imitation Learning
    - Implement and test out simple imitation learning algorithms in simulation
  - HW 2 – Model-free RL
    - Implement and test out policy gradient and actor critic methods
  - HW 3 – Model-based RL
    - Implement and test out model-based RL algorithms
- Submit through canvas with a small written report.
Who am I?

- Assistant professor in CSE
- Grew up in Oregon/India, last 10 years in Berkeley
- Undergrad Berkeley, Ph.D. Berkeley, Postdoc MIT.
- Interests: RL/robotics/optimization and control/robustness and generalization
- Outside of work: Tennis/soccer/sketching/dog enthusiast
Who is Patrick?

1. First-year PhD in Robotics/ML
2. Life Trajectory: Grew up in the bay area, did my undergrad at Berkeley
3. Research Interests: Robot Learning, Reinforcement Learning, Manipulation
4. Outside work: baking, guitar, tennis/pickleball, hiking, gyming, reading
Who is Zoey?

- Final Year PhD in Robotics
- Life Trajectory: grow up in China, last 10 years: Seattle-Japan-Switzerland-Seattle
- Academic Trajectory: Optics & Lasers -> Medical Image Analysis -> Robotics
- Research Interest: Imitation Learning, Robot Perception, Manipulation
- Outside work: piano, painting, board game, hiking, dog owner in 3 weeks!
What is this course about?

- The design and **practice** of reinforcement learning algorithms
What is this course about?

- Building RL algorithms that are practical for real applications
  - Sample efficient
  - Operates from high-dimensional observations
  - Continually improving

RL algorithms were not conceived to operate under practical assumptions, needs some extra work.
What is this course about?

- Practice implementing and **tuning** sequential decision making algorithms

- **Imitation learning**

- **Model-Free RL**

- **Model-Based RL**

- Most RL algorithms require tips and tricks, we will study them
What is this course not about?

- Not a pure theory course, more an applied-RL course
  - For pure theory classes, recommend CSE 541 or Simon’s CSE 542

- Only cover the theory needed to derive algorithms

Lemma 2.10. Let $\delta > 0$. With probability greater than $1 - \delta$,

$$|(P - \hat{P})V^*| \leq \sqrt{\frac{2\log(2|S||A|/\delta)}{N}} \sqrt{\text{Var}_P(V^*)} + \frac{1}{1 - \gamma} \frac{2\log(2|S||A|/\delta)}{3N}.$$ 

Theorem 4.3 (FQI guarantee). Fix $K \in \mathbb{N}^+$. Fitted $Q$ iteration guarantees that with probability $1 - \delta$,

$$V^* - V_{\pi^K} \leq \frac{1}{(1 - \gamma)^2} \left( \sqrt{\frac{22C\nu^2\ln(|J|2K)}{n}} + \sqrt{20C\epsilon_{\text{approx},\nu}} \right) + \frac{\gamma^KV_{\max}}{(1 - \gamma)}.$$
What should we be able to do post CSE 542?
Course logistics and scope

What is RL, a formal definition

Why should we care?

Going beyond RL
Ok so let’s try and define Reinforcement Learning

Using trial and error in an environment to learn a strategy to maximize some notion of “reward”

Can learn in arbitrary settings

Can generalize to new states

No expert corpus needed

Go from expert label → scalar measure of success

Easy (?) way for agents to continue improving their own behavior on deployment
Hypothesis: By designing algorithms that can improve themselves, we can reach fully intelligent systems.

"Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain" – Alan Turing

Rather than try to directly replicate behaviors, try to replicate adaptive learning mechanisms.
Why reinforcement learning?: Practical

A useful tool for building continually improving robots!

Robot learning is different for several reasons:
1. Sequential decision making problem (Non IID)
2. Large amounts of expert robot data may be expensive
3. Naturally multi-task and continual
4. Behaviors may be hard to pre-program

Robots that collect their own data to improve!
A Little History on Reinforcement Learning

Two distinct threads converged to give rise to modern RL

Animal Psychology

Optimal Control

Ideas from temporal difference learning/dynamic programming united these fields!

\[
\min_{x,u} \int_0^x L(t,x(t),u(t)) \, dx \\
\text{w.r.t.} \\
x'(t) = f(x(t),u(t))
\]

Sutton and Barto
A Little History on Reinforcement Learning

Klopf/Sutton/Barto brought together ideas from psych/neuro and computational TD learning

Harry Klopf

Introduced the ideas of “generalized reinforcement” – linked together TD learning and trial and error learning from psychology

Brain Function and Adaptive Systems – A Heterostatic Theory
A. Harry Klopf

Also had major contributions from Watkins, Shannon, Minsky, Tesauro, Michie, Sutton, Samuel, etc!
Some evidence about RL in the brain

Reinforcement learning in the brain

Yael Niv
Psychology Department & Princeton Neuroscience Institute, Princeton University

Shows the importance of temporal difference reward prediction error in processes in the brain

Dopamine ≠ reward, rather dopamine corresponds strongly to errors in long term reward prediction (aka TD errors) (Montague ‘96, Schultz ‘97). Some inconsistencies, e.g. Dealing with aversive events like pain

Likely much more research needed, since decisions can be made in the absence of dopamine → multiple different RL processes in the brain
Playing Atari with Deep Reinforcement Learning

Volodymyr Mnih  Koray Kavukcuoglu  David Silver  Alex Graves  Ioannis Antonoglou
Daan Wierstra  Martin Riedmiller
DeepMind Technologies
A Little History on Modern Reinforcement Learning (my view)

Trust Region Policy Optimization

John Schulman  
Sergey Levine  
Philip Moritz  
Michael Jordan  
Pieter Abbeel

University of California, Berkeley, Department of Electrical Engineering and Computer Sciences

Iteration 0
Since then, we have gotten RL now to power a variety of high-impact applications
Let’s define a formalism

Probability theory

Optimization

Machine Learning
Preliminaries: Markov Chains

Future is independent of past, conditioned on the present

Goal of Markov chain: running the Markov chain leads to sampling from stationary distribution $d^\pi$

Balance equation

$$Td^\pi = d^\pi$$

Useful in sampling based inference eg MCMC

How can we make this useful for decision making?
Augment Markov chain with rewards and actions

States: $S$  
Initial state dist: $\rho_0(s)$

Actions: $A$  
Discount: $\gamma$

Rewards: $R$

Transition Dynamics - $p(s_{t+1}|s_t, a_t)$

Markov property  
$p(s_1, s_2, s_3) = p(s_3|s_2)p(s_2|s_1)p(s_1)$

Trajectory  
$\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T)$
Task: Place kettle in sink

State: Camera Images / Joint Encoders

Action: Joint torques/velocities

Reward: Distance from kettle to sink

Transition: World physics
Aside: Partially Observed MDPs

Not every environment is an MDP, in-fact most are POMDPs

POMDPs are hard, we will try to avoid them!
Reinforcement Learning Formalism

Maximize the sum of expected rewards under policy

Needs to be learned
Reinforcement Learning Formalism

Rules for choosing actions

Policy $\pi$

$S_t$  $a_t$  $S_t$

needs to be learned

Trajectory sampled using policy

$\max_{\theta} \mathbb{E}_{T \sim \pi_\theta} \left[ \sum_{t=0}^{T} r(s_t, a_t) \right]$
Main thing to learn - Policies

Policies are **mappings** from states to distributions over actions.

\[
\pi(a|s) = \langle \phi(s, a), w \rangle
\]
Main thing to learn - Policies

Policies are mappings from states to **distributions** over actions

- **Gaussian**: $\mu(s), \Sigma(s)$
- **Categorical**: $p_1(s), p_2(s), \ldots, p_k(s)$
- **Mixture of Gaussians**: $\mu_1(s), \Sigma_1(s), w_1$
  $\mu_2(s), \Sigma_2(s), w_2$
  $\ldots$
  $\mu_N(s), \Sigma_N(s), w_N$
- **Diffusion Models**: $\nabla_a \log \pi(a|s)$
Let's revisit policies: stochastic vs deterministic

Lemma 1: Every MDP has at least one optimal *deterministic* policy

**Theorem 1.7.** Let $\Pi$ be the set of all non-stationary and randomized policies. Define:

$$V^*(s) := \sup_{\pi \in \Pi} V^\pi(s)$$

$$Q^*(s, a) := \sup_{\pi \in \Pi} Q^\pi(s, a).$$

which is finite since $V^\pi(s)$ and $Q^\pi(s, a)$ are bounded between 0 and $1/(1 - \gamma)$.

There exists a stationary and deterministic policy $\pi$ such that for all $s \in S$ and $a \in A$,

$$V^\pi(s) = V^*(s)$$

$$Q^\pi(s, a) = Q^*(s, a).$$

We refer to such a $\pi$ as an optimal policy.

Intuition: pick the best possible action at every state

Stochastic policies will help in the search/optimization process for finding (close to) deterministic policies.
Let’s take a closer look at the objective: horizon

\[
\max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[ \sum_{t=0}^{T} r(s_t, a_t) \right]
\]

Finite horizon

\[
\mathbb{E}_{\pi^t_\theta} \left[ \sum_{t=0}^{T} r(s_t, a_t) \right]
\]

Time-dependent policy (not stationary)

\[
\mathbb{E}_{\pi^t_\theta} \left[ \sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right]
\]

Infinite horizon discounted

\[
\mathbb{E}_{\pi_\theta} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]
\]

Time-independent (stationary) policy

\[ \Rightarrow \text{Need discount to prevent blow up} \]

**Lemma:** there always exists a stationary optimal policy
Unpacking the Expectation

\[
\max_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}} \left[ \sum_{t=0}^{T} r(s_t, a_t) \right]
\]

\[
\mathbb{E}_{\pi_{\theta}^{t}} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]
\]

**Trajectory View** - Ancestral sampling along MDP

- Initial state
- Policy
- Dynamics

\[
\mathbb{E} \left[ s_0 \sim \rho_0(s) \right. \\
\left. a_0 \sim \pi_{\theta}(\cdot|s_0) \right. \\
\left. s_1 \sim p(\cdot|s_0, a_0) \right. \\
\left. a_1 \sim \pi_{\theta}(\cdot|s_1) \right. \\
\left. s_2 \sim p(\cdot|s_1, a_1) \right. \\
\ldots
\]

\[
\sum_{t=0}^{T} r(s_t, a_t)
\]

**Stationary View** – sampling from stationary dist

\[
d_\pi^t(s, a) = \mathbb{P}(s_t = s, a_t = a \mid s_0 \sim \rho_0, \forall i < t, a_i \sim \pi_{\theta}(\cdot|s_i), s_{i+1} \sim p(\cdot|s_i, a_i))
\]

(Likelihood of being at state s, action a at time step t)

\[
\mu_{\pi}^{\gamma}(s, a) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t d_\pi^t(s, a)
\]

(Likelihood of being at state s, action a across all steps)

\[
\gamma \text{ subsumed into } \mathbb{E} \quad \mathbb{E}_{(s, a) \sim \mu_{\pi}^{\gamma}(s, a)} \left[ r(s, a) \right]
\]

No sequential sampling

No sum over rewards
Some notation: $Q$-functions and $V$-functions

Estimate of how “good” a policy is – estimate of future returns under a policy $\pi$

**Q-function**

Take one action and then follow policy from $s$

$$Q^\pi(s, a) = \mathbb{E}_{\pi,p} \left[ \sum_t r(s_t, a_t) \mid s_0 = s, a_0 = a \right]$$

**V-function**

Follow policy from $s$

$$V^\pi(s, a) = \mathbb{E}_{\pi,p} \left[ \sum_t r(s_t, a_t) \mid s_0 = s \right]$$

$$V^\pi(s, a) = \mathbb{E}_{a \sim \pi(\cdot|s)} [Q^\pi(s, a)]$$

$J(\pi) = \mathbb{E}_{s \sim \rho_0(s)} [V^\pi(s)]$

Average value over initial states

Will be useful soon!
Ok so where does deep learning fit in?

Avoids expensive hand-design for adaptive agents, learn end-to-end: sensors → actions

Policies/Q-values/model are represented as deep neural networks

Leads to non-trivial challenges in learning and optimization!
Why is deep RL important now?

Deep models have enabled the huge advances in modern AI.

We are betting that the same holds for RL.
Ok so is this just supervised learning?

Supervised learning aims to maximize likelihood of observed data under the model

$$\max_\theta \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[ \log \hat{p}_\theta(y|x) \right]$$
Why is this not just supervised learning?

Supervised Learning

\[
\max_{\theta} \mathbb{E}_{(x,y) \sim D} [\log \hat{p}_\theta(y|x)]
\]

Sampling from expert

\[ D_{KL}(p^* || p_\theta) \quad \text{IID} \]

Reinforcement Learning

\[
\max_{\theta} \mathbb{E}_{\tau \sim \pi_\theta} \left[ \sum_{t=0}^{T} r(s_t, a_t) \right]
\]

Sampling from policy

\[ D_{KL}(p_\theta || p^*) \quad \text{Non-IID} \]
Why is this not just supervised learning?

**Supervised Learning**

$$\max_\theta \mathbb{E}_{(x,y) \sim D} \left[ \log \hat{p}_\theta(y|x) \right]$$

**Reinforcement Learning**

$$\max_\theta \mathbb{E}_{\tau \sim \pi_\theta} \left[ \sum_{t=0}^T r(s_t, a_t) \right]$$

The resulting paradigms are different in many ways:

1. Optimization and learning dynamics
2. Balancing exploration and exploitation

But many overlapping tools! In fact often we try to convert RL into a supervised problem.
Lecture Outline

Course logistics and scope

What is RL, a formal definition

Why should we care?

Going beyond RL
Ok so why should we care about RL?

- Solves sequential decision making problems
- Enables continual improvement
- Has black-box assumptions
- Reduces burden of human data collection
Applications of RL: Robotics

RL can enable robotic learning of hard to specify/script behaviors in the presence of contact.
Applications of RL: Large Language Models

Systematically finds and reduces model hallucinations using RLHF

---

**Step 1**
**Collect demonstration data, and train a supervised policy.**

- A prompt is sampled from our prompt dataset.
- A labeler demonstrates the desired output behavior.
- This data is used to fine-tune GPT-3 with supervised learning.

**Step 2**
**Collect comparison data, and train a reward model.**

- A prompt and several model outputs are sampled.
- A labeler ranks the outputs from best to worst.
- This data is used to train our reward model.

**Step 3**
**Optimize a policy against the reward model using reinforcement learning.**

- A new prompt is sampled from the dataset.
- The policy generates an output.
- The reward model calculates a reward for the output.
- The reward is used to update the policy using PPO.
Applications of RL: Games

Both single and multi-agent RL has proven transformative for game AI

Particularly well suited to RL assumptions
Applications of RL: Science and Engineering

RL has started to become a useful tool for engineering design

- Chip Design
- Weather balloon navigation
- Crystal design
Hypothesis: Intelligence with and without embodiment looks drastically different

Elephants don’t play chess!
Why must we study RL in the real world?

Hypothesis: Agents that learn with embodiment will have emergent complexity in complex, dynamic environments

Increasingly interesting behavior
Where is Reinforcement Learning not useful?

Not the right call for very safety-critical, repetitive applications
Where is Reinforcement Learning “potentially” useful?

Domains which have high diversity, yet relatively cheap autonomous data collection

But these domains are not as simple as just running RL algorithms!
Lecture Outline

Course logistics and scope

What is RL, a formal definition

Why should we care?

Going beyond RL
So is sequential decision making = RL?

We conflated sequential decision making and RL!

RL is sequential decision making under a particular set of assumptions:
1. Sampling access to the environment
2. Access to reward
3. Goal-directed behavior

Trajectory optimization/planning

Imitation Learning

Unsupervised Decision Making

Interleaving Graph Search and Trajectory Optimization for Aggressive Quadrotor Flight
Ran Kumar Natarajan, Howie Choset and Maxim Likhachev
Trajectory Optimization

Sequential decision making with “known” models

We combine RRT and local smoothing of contact dynamics to generate complex contact-rich manipulation plans.

Interleaving Graph Search and Trajectory Optimization for Aggressive Quadrotor Flight

Ramkumar Natarajan, Howie Choset and Maxim Likhachev

May be hard to construct perfect, known models
Imitation Learning

Sequential decision making provided expert data

Often called learning from demonstrations
Self-Supervised Prediction of the World

Sequential decision making without reward – self-supervised prediction

Generate a playable world set in a futuristic city

Often called model-based RL
How should we think about designing effective RL algorithms?

Stable performant optimization algorithms

Easy to specify objectives

Efficient data collection
Class Structure

Model-free Reinforcement Learning
- Policy Gradient
- ADP

Model-based Reinforcement Learning

Unifying Perspectives on RL and IRL

Frontiers
- Exploration
- Learning from Prior Data
- Learning across tasks