Deep Reinforcement Learning

CS 224R

Welcome!

Introductions

Prof. Chelsea Finn

Instructor

Amelie Byun

Course Manager

Marcel Torne

Head Teaching Assistant

Anushree Aggarwal

Abhijnya Bhat

Shengqu Cai

Rahul Chand

Perry Dong

Max Du

Tian Gao

Joy He-Yueya

Ethan Hsu

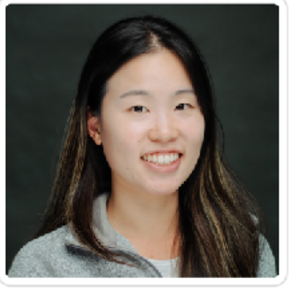
Joonwon Kang

Yash Kankariya

Riya Karumanchi

Tyler Lum

Anubha Mahajan



Alex Nam

Ifdita Orney

Anikait Singh

Megha Srivastava

Ke Wang

Jonathan Yang

The Plan for Today

1. Course goals & logistics
2. Why study deep reinforcement learning?
3. Intro to modeling behavior and reinforcement learning

Key learnings goals:

- how to represent behavior
- how to formulate a reinforcement learning problem
- basics of imitation learning

Information & Resources

Course website: http://cs224r.stanford.edu/ ← We have put a lot of info here. Please read it. :)

Ed, Gradescope: Connected to Canvas

Staff mailing list: cs224r-staff-spr2526@cs.stanford.edu ← Student liaison, course manager, head CA, me

Office hours: Course website & Canvas, start today.

OAE letters can be sent to staff mailing list or in private Ed post.

Lectures & Office Hours

Lectures

- In-person, livestreamed, & recorded
- Aiming to make it interactive. I will ask you questions. Ask me questions too!
- A few guest lectures (Noam Brown from OpenAI, Archit Sharma from Google DeepMind, Guanya Shi from Amazon FAR/CMU)

Office hours

- mix of in-person and virtual

What do we mean by deep reinforcement learning?

Sequential decision-making problems

A system needs to make *multiple* decisions based on stream of information.

observe, take action, observe, take action, ...

AND the solutions to such problems

- imitation learning
- model-free & model-based RL
- offline & online RL
- multi-task & meta RL
- RL for LLMs
- RL for robots

and more!

Emphasis on solutions that scale to deep neural networks

How does deep RL differ from other ML topics?

Supervised learning

Given labeled data: $\{(x_i, y_i)\}$, learn $f(x) \approx y$

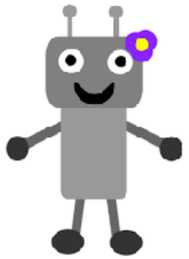
- directly told what to output
- inputs x are independently, identically distributed (i.i.d.)

Reinforcement learning

Learn *behavior* $\pi(a \mid s)$.

- from experience, indirect feedback
- data **not** i.i.d.: actions a affect the future observations.

Behavior can include:



motor control

chat bots

game playing

driving

web agents

We can’t cover everything in deep RL.

We’ll focus on:

- core concepts behind deep RL methods
- implementation of algorithms
- examples in robotics, control, language models (but techniques generalize broadly)
- topics that we think are most useful & exciting!

For more theory & other applications, see CS234!

Core goal: Able to understand and implement existing and emerging methods.

Pre-Requisites

Machine learning: CS229 or equivalent.

e.g. we'll assume knowledge of SGD, cross-val, calculus, probability theory

Some familiarity with deep learning:

- We'll build on concepts like backpropagation, neural networks, sequence models
- Assignments will require training networks in PyTorch.
- Megha will hold a PyTorch review session on Friday, 3:30 pm in Skilling Auditorium.

Some familiarity with reinforcement learning:

- We will go quickly over the basics.
- See Sutton & Barto or CS 221 for intro RL content

Coursework and Grading

- **3x assignments:** HW1 worth 10%, HW2 and HW3 worth 15% each
- **Final default or custom course project** (1-3 people, 35%)
  - proposal (4%), milestone (5%), poster (8%), report (18%)
- **Mid-term**: May 15th during class, worth 25%, covers content up to week 6
- **Late days**: 5 free late days; maximum of 2 free late days per assignment
- **Collaboration & AI tools**
  - *Write solutions on your own*
  - Please read course website, honor code, AI tools policy
  - Document collaborators & use of AI tools.

Coursework

Homeworks: Implement different methods in PyTorch, run experiments in physics simulators, navigation environments

Homework 1: Imitation learning

Homework 2: Online reinforcement learning

Homework 3: Offline reinforcement learning

Due on Fridays at 9 pm PT.

Project:

- Custom project - propose your own topic, or
- Default project - fine-tune an LLM with RL + open-ended extension
- Teams of 1-3 students, encouraged to use your research if applicable

A bit of advice

Deep RL methods take time to learn behavior!

We try to make homeworks fast to train.
(e.g. by using simple environments)

But, they will still take some time & you may choose to be more ambitious in your project.

We recommend that you don’t start HWs/project deliverables the night before the deadline. :)

One more thing

We have been working hard to develop a great course!

But, we will probably make mistakes.

We would **love** your feedback both for this iteration & future iterations.

—> high-resolution feedback form sent weekly to subset of students.

Initial Steps

1. Homework 1 coming out soon — due Fri 4/10 at 9:00 pm PT
2. Start forming final project groups if you want to work in a group

The Plan for Today

1. Course goals & logistics
2. **Why study deep reinforcement learning?**
3. Intro to modeling behavior and reinforcement learning

Why study deep reinforcement learning?

1. Going **beyond supervised (x, y) examples**
   - AI model predictions have consequences!
   - When direct supervision isn't available

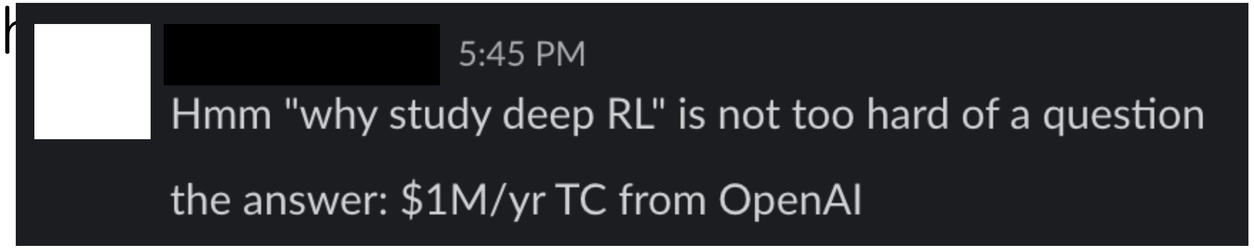
How can we take them into account?

Learn from *any* objective.

2. **Widely used and deployed** for performant AI systems

3. Learning from experience seems **fundamental to intelligence**
   - RL can **discover new solutions**

4. Plenty of exciting open research



Why study deep reinforcement learning?

Beyond supervised learning from (x, y) examples

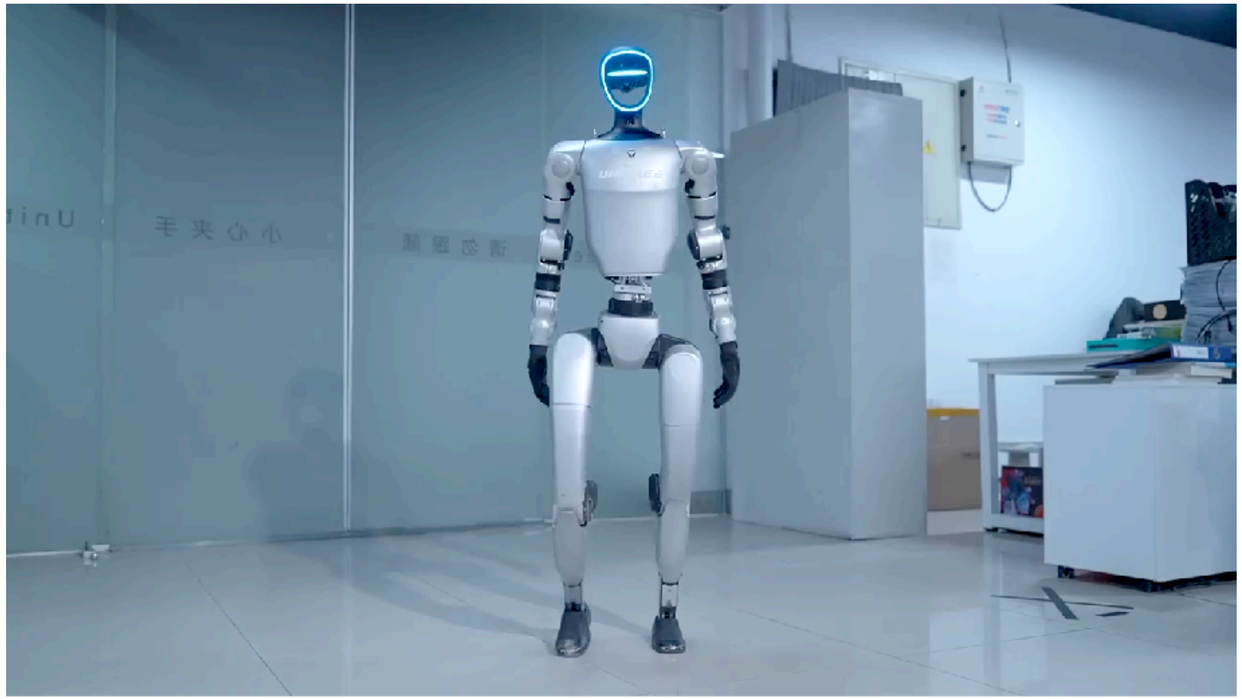
Decision-making problems are everywhere!

a. Any sort of AI agent: robots, autonomous vehicles, web assistants
b. What if you want your AI system to interact with people? chatbots, recommenders
c. What if deploying your system affects future outcomes & observations? "feedback loops"
d. What if don't have labels or your objective isn't just accuracy? (and isn't differentiable!)

Why study deep reinforcement learning?

Widely used for performant AI systems

Learning complex physical tasks: legged robots



Unitree

Why study deep reinforcement learning?

Widely used for performant AI systems

Learning complex physical tasks: robot manipulation

Physical Intelligence $\pi_{0.6}$

Sunday Robotics ACT-1

Why study deep reinforcement learning?

Widely used for performant AI systems

Learning to play complex games

Ability to **discover** new solutions:

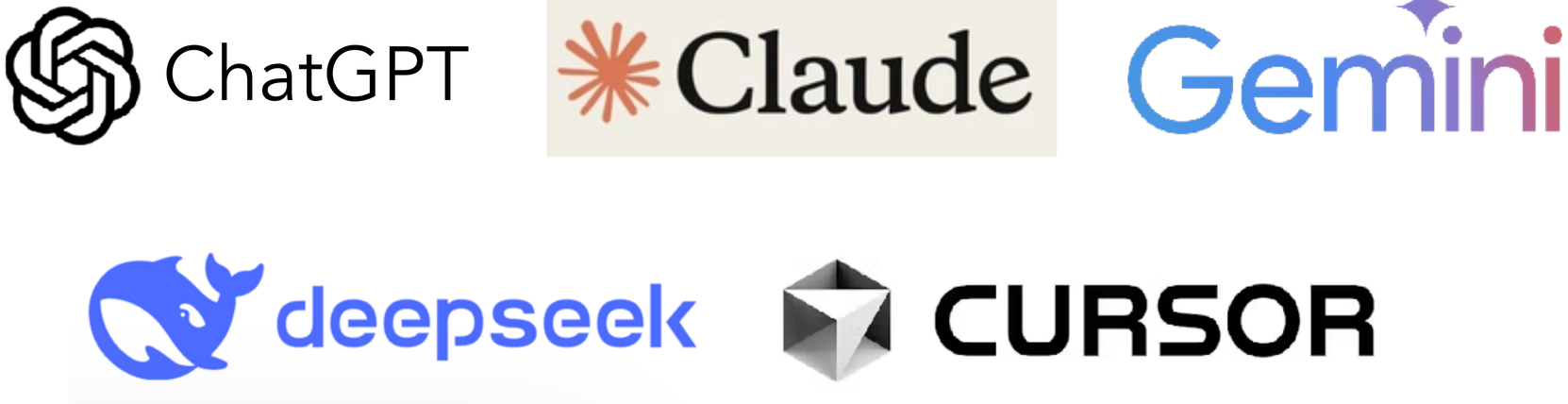
“Move 37” in Lee Sedol AlphaGo match surprises everyone

Why study deep reinforcement learning?

Widely used for performant AI systems

Not just robots and games!

Nearly all modern language models use some form of RL for post-training.



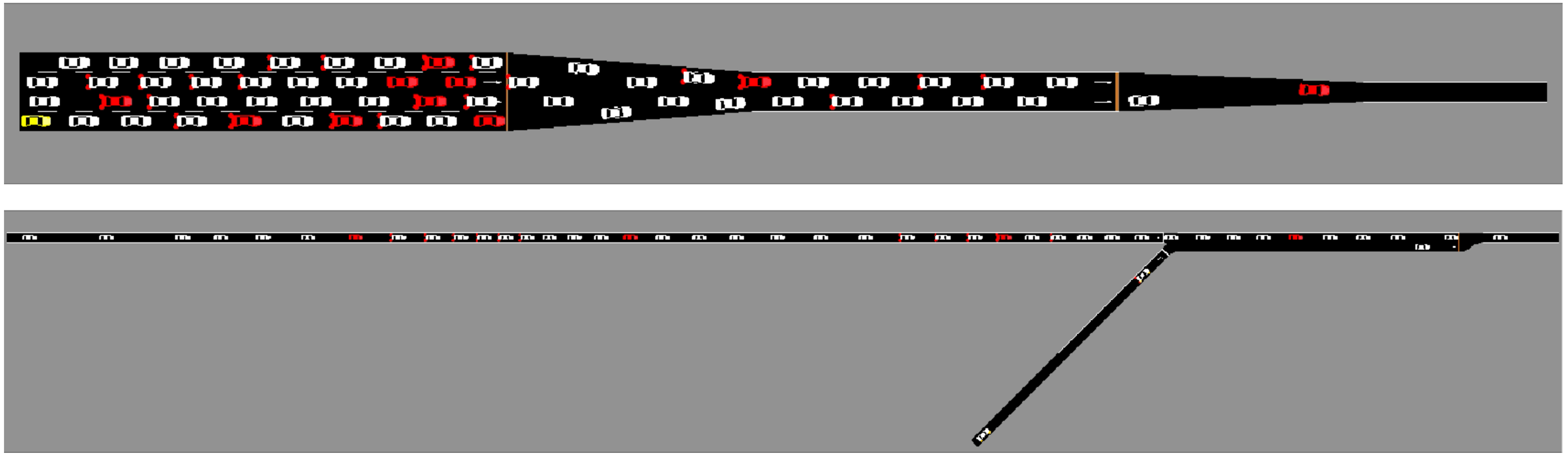
especially for more advanced reasoning.

Why study deep reinforcement learning?

Widely used for performant AI systems

Not just robots and games!

Research on traffic control



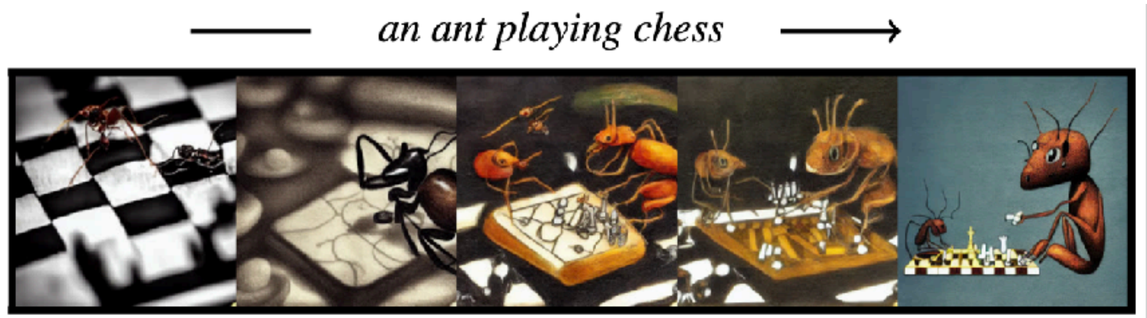
Source: https://mit-wu-lab.github.io/automatic_vehicular_control/

Why study deep reinforcement learning?

Widely used for performant AI systems

Not just robots and games!

Training generative image models to follow their prompt



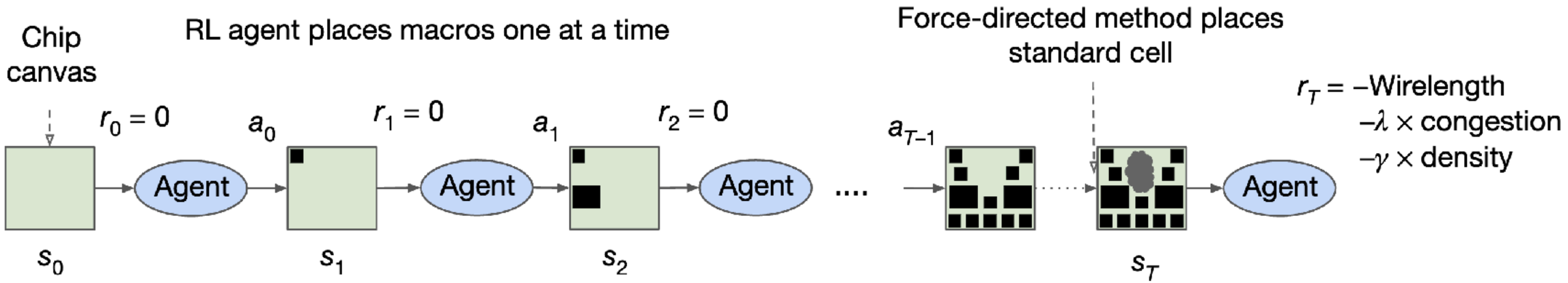
Source: https://rl-diffusion.github.io/

Why study deep reinforcement learning?

Widely used for performant AI systems

Not just robots and games!

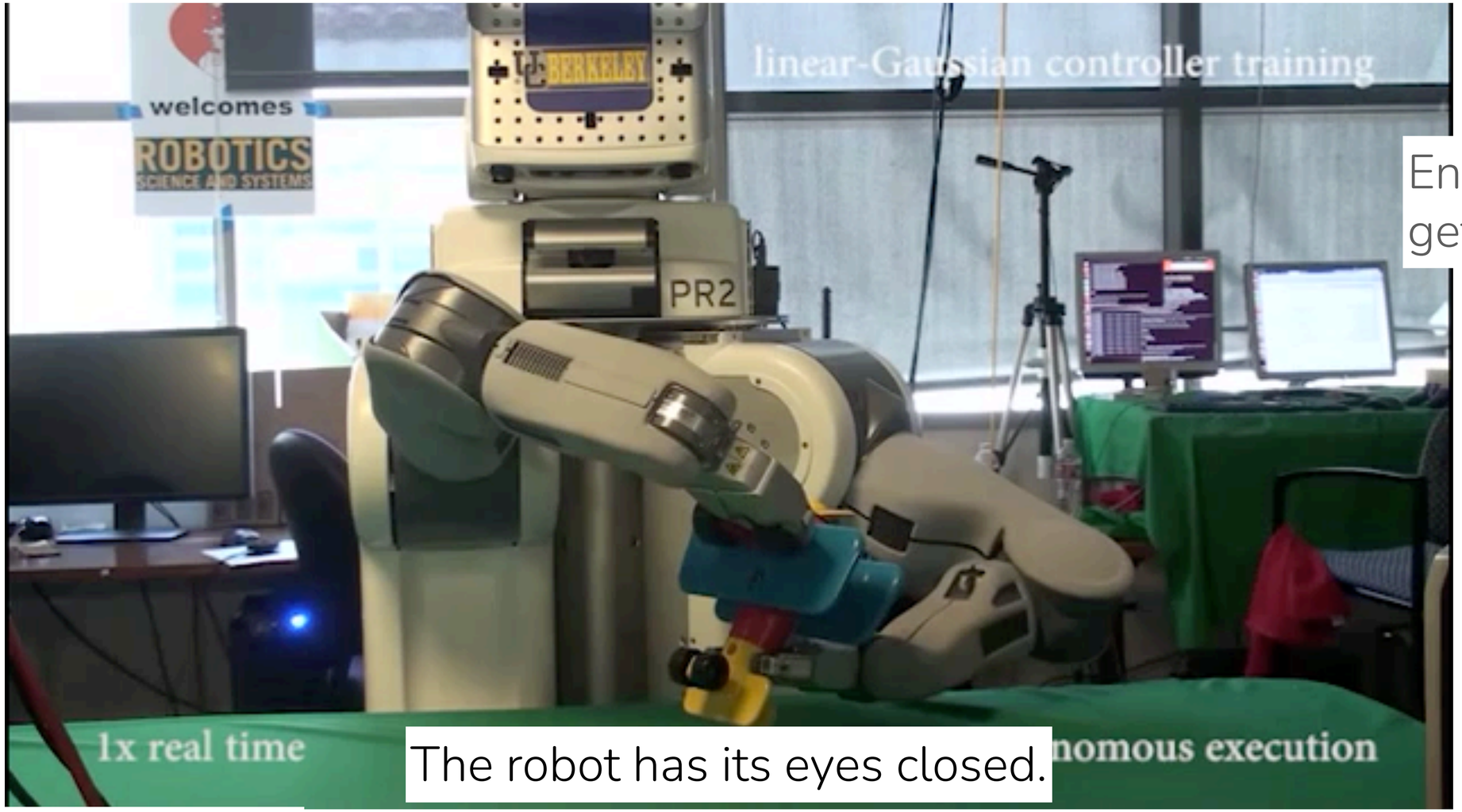
Chip design, in Google's production TPU chips



Source: https://research.google/blog/chip-design-with-deep-reinforcement-learning/

Why study deep reinforcement learning?

Fundamental aspect of intelligence



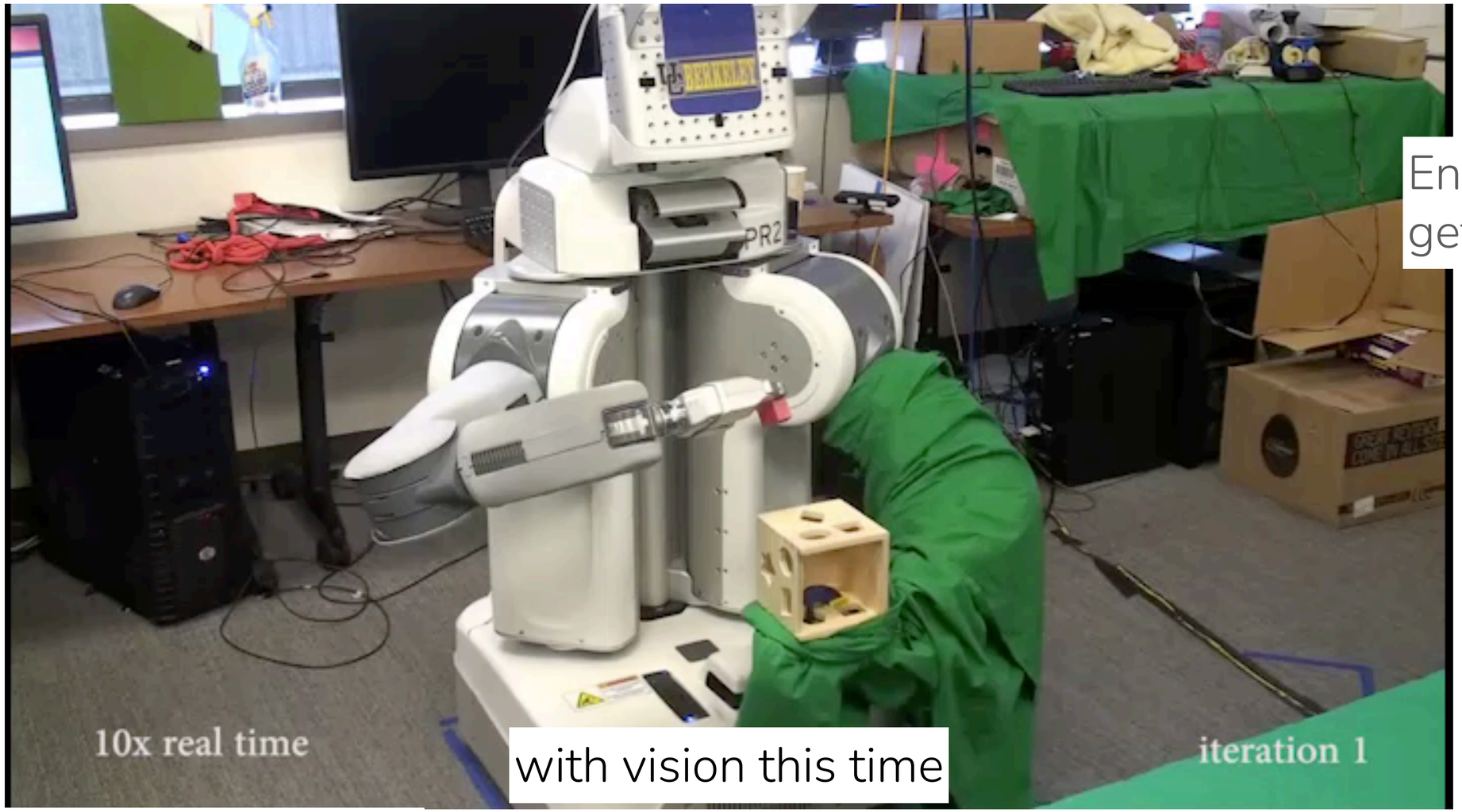
Enables the ability to get better with practice

The robot has its eyes closed.

Levine et al. ICRA ‘15

Why study deep reinforcement learning?

Fundamental aspect of intelligence

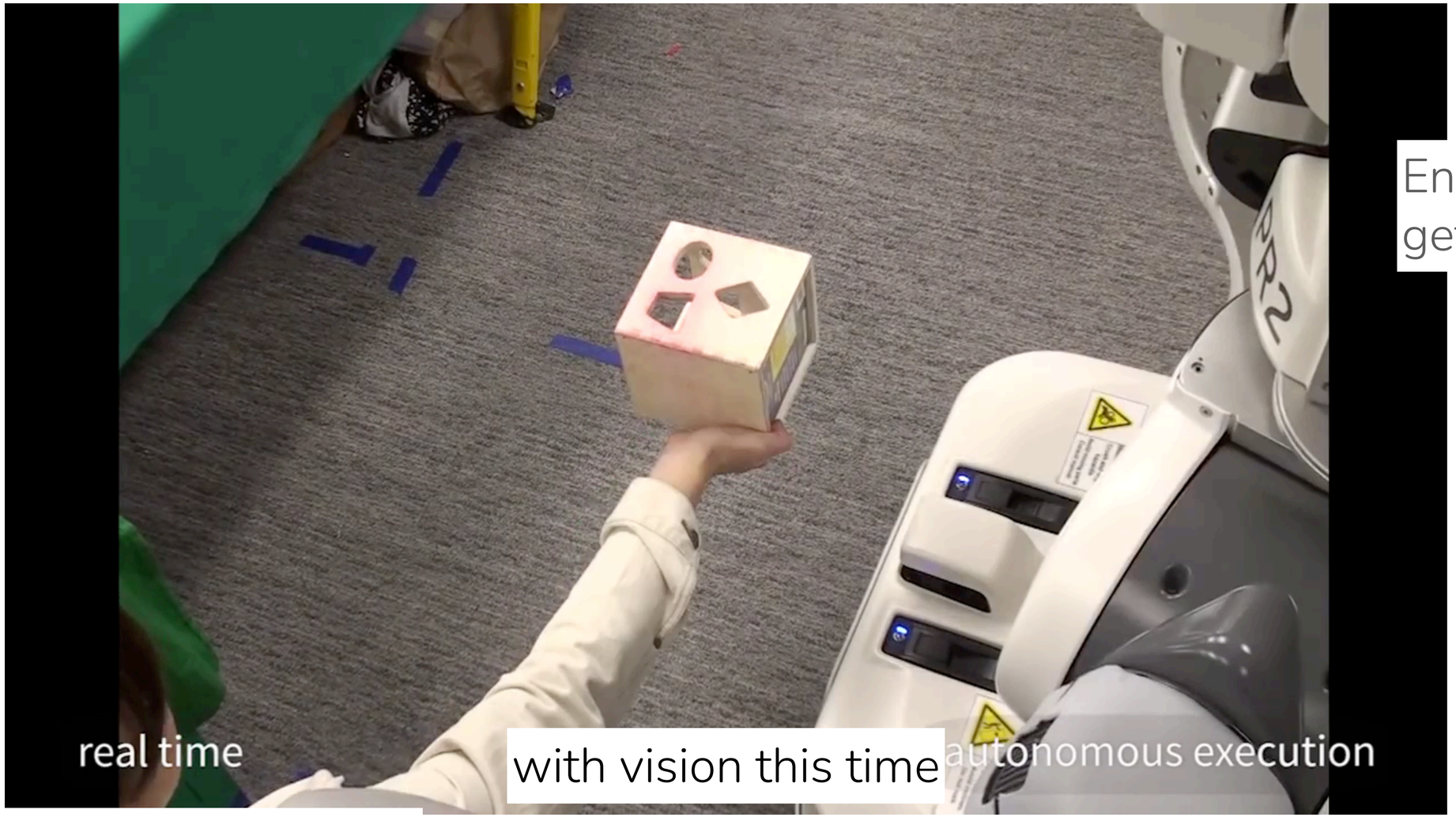


Enables the ability to get better with practice

Levine*, Finn* et al. JMLR ‘16

Why study deep reinforcement learning?

Fundamental aspect of intelligence



Enables the ability to get better with practice

Levine*, Finn* et al. JMLR ‘16

Why study deep reinforcement learning?

Still lots of exciting research problems!

How does robot learn to represent what is good or bad for the task?

How can an agent generalize its behavior to many different scenarios?

(Can we apply such a system at scale?)

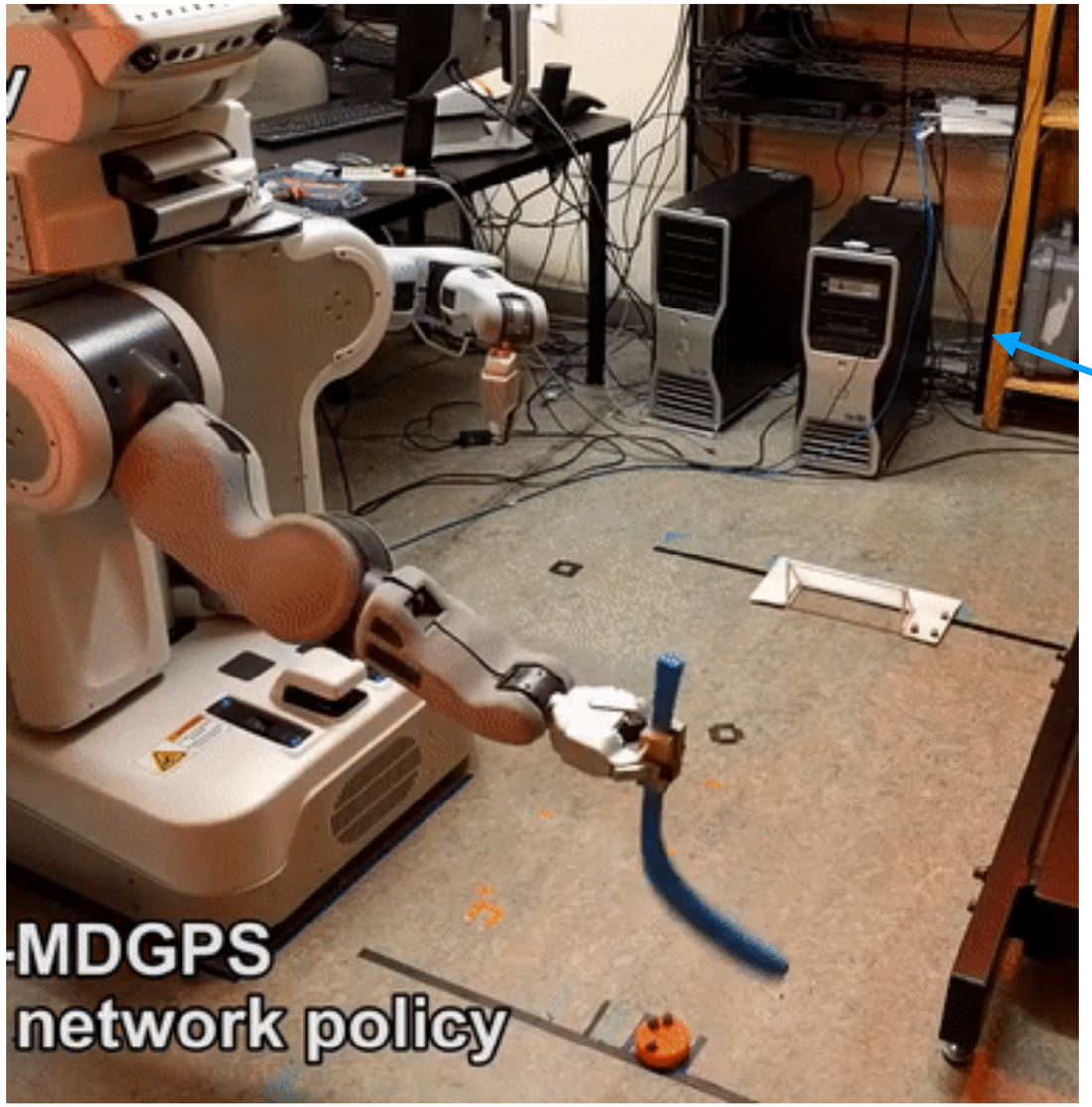
Leverage large, diverse datasets

Transfer from other tasks, goals

Can use RL to learn long-horizon tasks, like cooking a meal?

Can robots practice fully autonomously?

Behind the scenes of RL…



Yevgen

Yevgen is doing more work than the robot!

It's not practical to collect a lot of data this way.

Why study deep reinforcement learning?

Still lots of exciting research problems!

How does robot learn to represent what is good or bad for the task? —> reward learning

How can an agent generalize its behavior to many different scenarios?

(Can we apply such a system at scale?)

Leverage large, diverse datasets —> offline RL

Transfer from other tasks, goals —> multitask RL, meta-RL

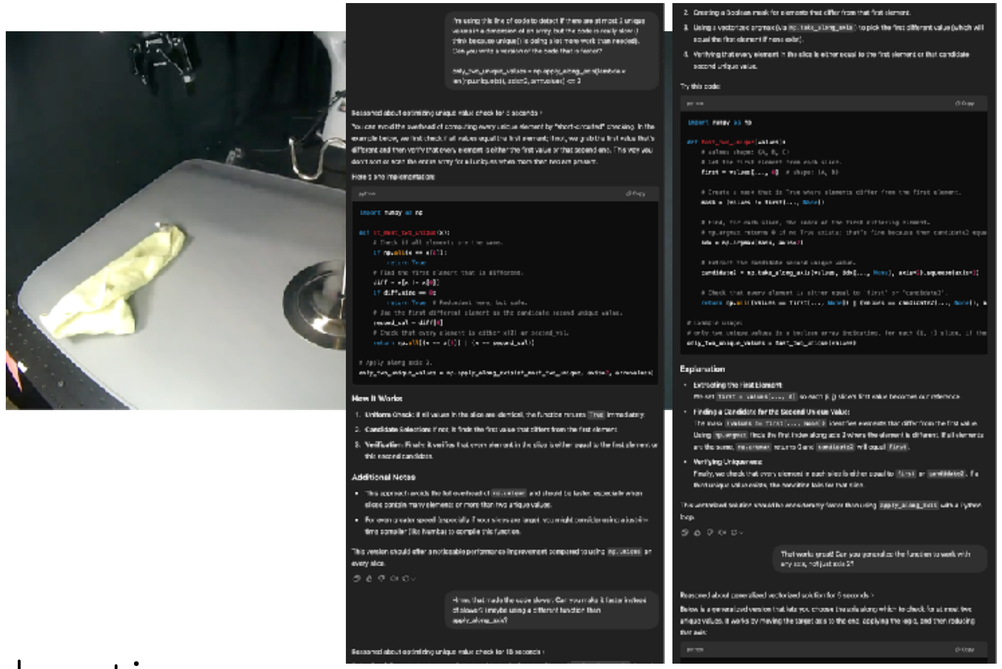
Can use RL to learn long-horizon tasks, like cooking a meal? —> hierarchy, reasoning

Can robots practice fully autonomously? —> RL for robots

The Plan for Today

1. Course goals & logistics
2. Why study deep reinforcement learning?
3. **Intro to modeling behavior and reinforcement learning**

How to represent experience as data?



state $\mathbf{s}_t$ - the state of the “world” at time t

action $\mathbf{a}_t$ - the decision taken at time t

trajectory $\boldsymbol{\tau}$ - sequence of states/observations and actions

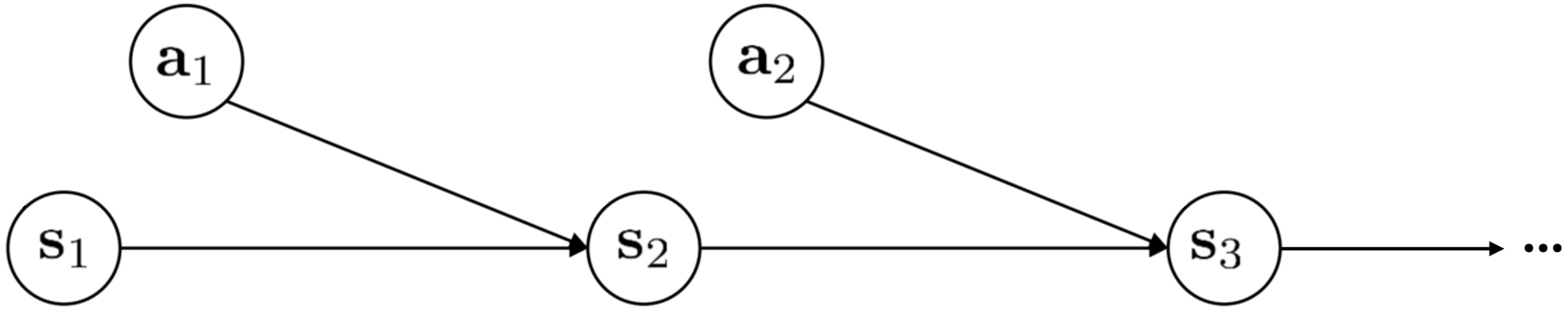
$$(\mathbf{s}_1, \mathbf{a}_1, \mathbf{s}_2, \mathbf{a}_2, \ldots, \mathbf{s}_T, \mathbf{a}_T)$$

(could be length T=1!)

reward function $\boldsymbol{r}(\mathbf{s}, \mathbf{a})$ - how good is $\mathbf{s}, \mathbf{a}$?

As a probabilistic model

Next state is purely a function of the current state and action (and randomness)

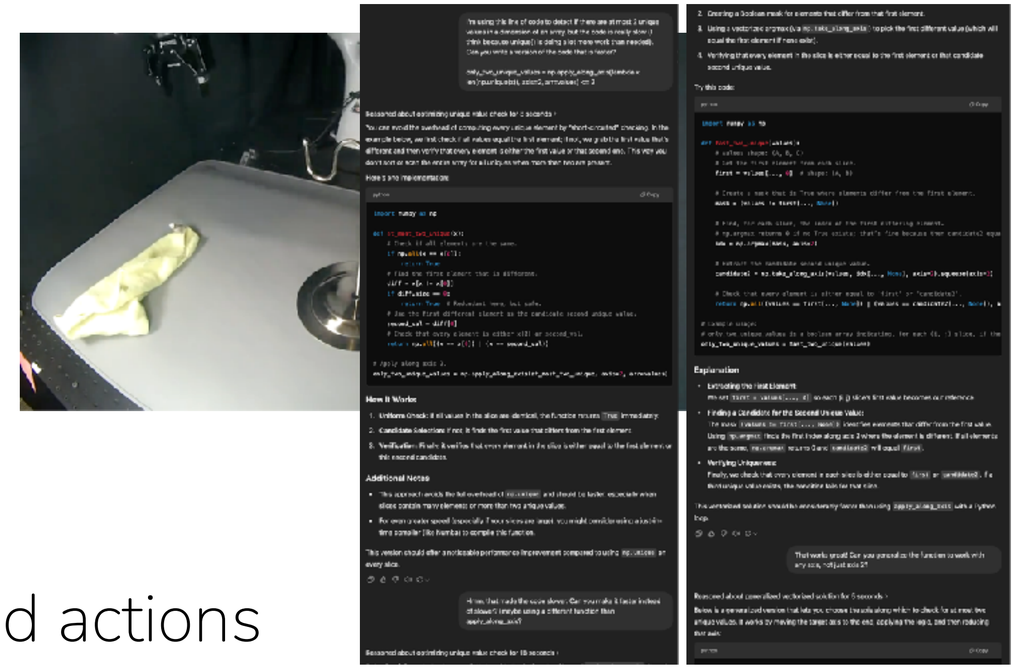


unknown dynamics $p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$

independent of $\mathbf{s}_{t-1}$

“Markov property”

How to represent experience as data?



state $\mathbf{s}_t$ - the state of the "world" at time t

action $\mathbf{a}_t$ - the decision taken at time t

trajectory τ - sequence of states/observations and actions

$$(\mathbf{s}_1, \mathbf{a}_1, \mathbf{s}_2, \mathbf{a}_2, \ldots, \mathbf{s}_T, \mathbf{a}_T)$$

(could be length T=1!)

reward function $r(\mathbf{s}, \mathbf{a})$ - how good is $\mathbf{s}, \mathbf{a}$?

*What if you don't have **direct access** to the state of the world?*

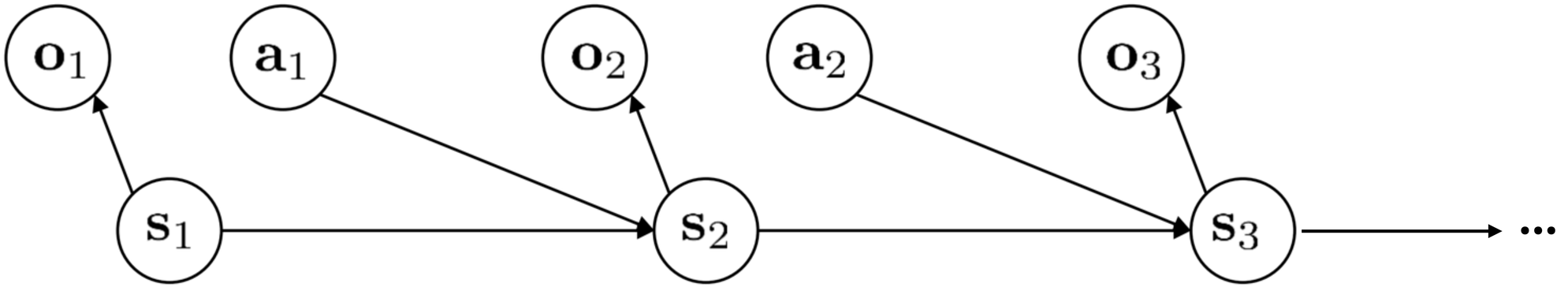
Option 1: Let your sensor readings be an approximation (e.g. camera images with good visibility are often close enough)

Option 2: Explicitly model partial observability

observation $\mathbf{o}_t$ - what the agent observes at time t

States vs. observations

Next state is purely a function of the current state and action (and randomness)



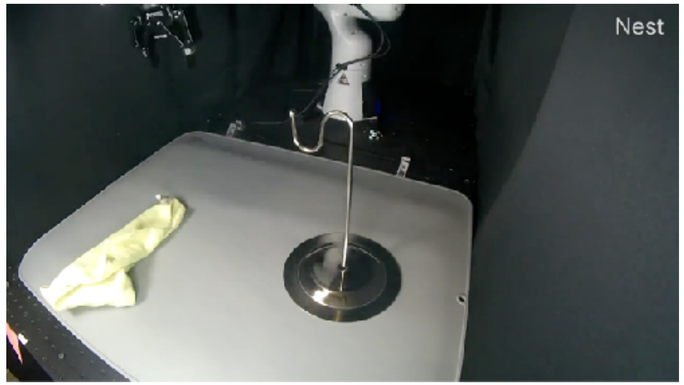
unknown dynamics $p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$

independent of $\mathbf{s}_{t-1}$

If you marginalize out states, future observations depend on all past observations.

i.e. $p(\mathbf{o}_{t+1} \mid \mathbf{o}_{1:t}, \mathbf{a}_{1:t}) \neq p(\mathbf{o}_{t+1} \mid \mathbf{o}_t, \mathbf{a}_t)$

Examples



state $\mathbf{s}$ - RGB images, joint positions, joint velocities

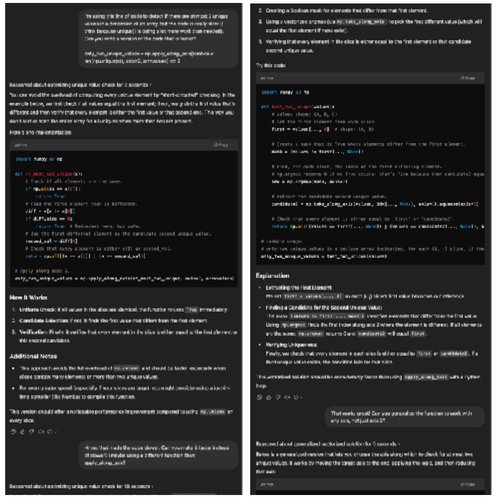
action $\mathbf{a}$ - commanded next joint position

trajectory $\boldsymbol{\tau}$ - 10-sec sequence of camera, joint readings, controls at 20 Hz

$$(\mathbf{s}_1, \mathbf{a}_1, \mathbf{s}_2, \mathbf{a}_2, \ldots, \mathbf{s}_T, \mathbf{a}_T),\ T = 200$$

reward $\boldsymbol{r}(\mathbf{s}, \mathbf{a})$ = 1 if the towel is on the hook in state $\mathbf{s}$

0 otherwise



observation $\mathbf{o}$ - the user's most recent message

action $\mathbf{a}$ - chatbot's next message

trajectory $\boldsymbol{\tau}$ - variable length conversation trace

$$(\mathbf{o}_1, \mathbf{a}_1, \mathbf{o}_2, \mathbf{a}_2, \ldots, \mathbf{o}_T, \mathbf{a}_T)$$

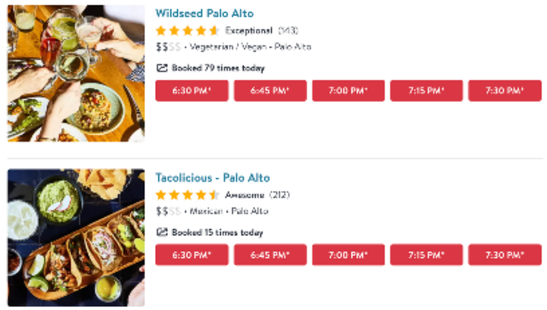
reward $\boldsymbol{r}(\mathbf{s}, \mathbf{a})$ = 1 if the user gives upvote

-10 if the user downvotes

0 if no user feedback

Think-pair-share: how to represent another example?

autonomous driving



web agent

poker player

choose your own!

Define

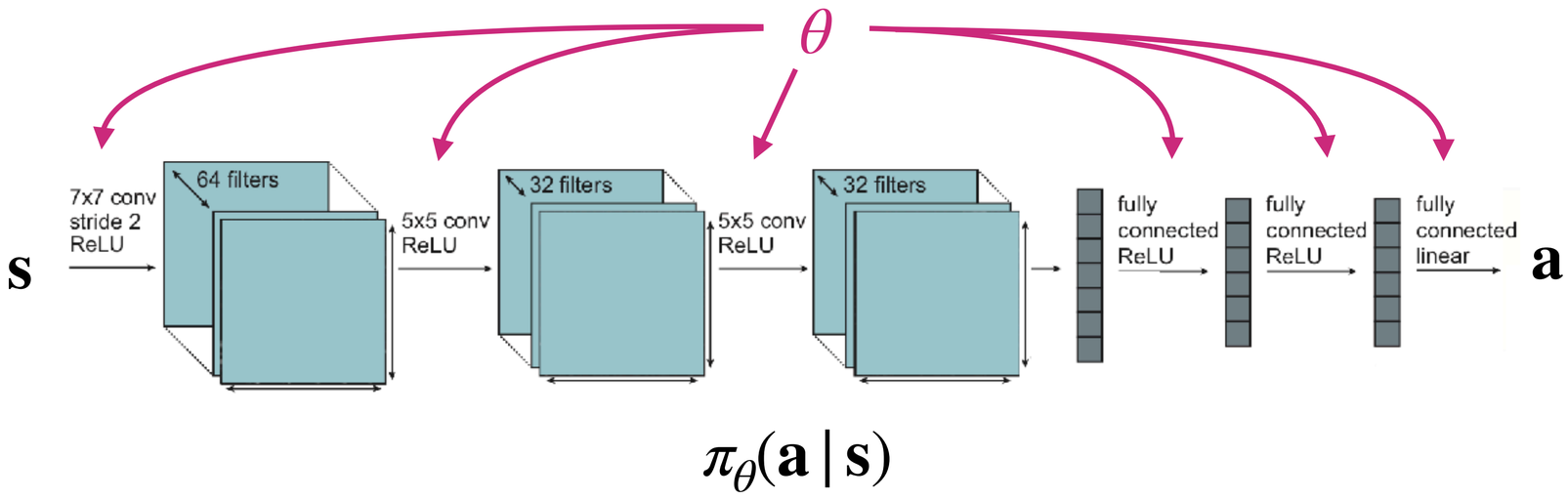
state $\mathbf{s}$ or *observation* $\mathbf{o}$

action $\mathbf{a}$

trajectory $\boldsymbol{\tau}$

reward $\boldsymbol{r}(\mathbf{s}, \mathbf{a})$

How to represent behavior with a neural network?



Observe state $\mathbf{s}_t$

Take action $\mathbf{a}_t$ (e.g. by sampling from *policy* $\pi_\theta(\cdot\,|\,\mathbf{s}_t)$)

Observe next state $\mathbf{s}_{t+1}$ sampled from unknown world dynamics $p(\cdot\,|\,\mathbf{s}_t, \mathbf{a}_t)$

Result: a *trajectory* $\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T$, also called a policy ***roll-out*** or an ***episode***

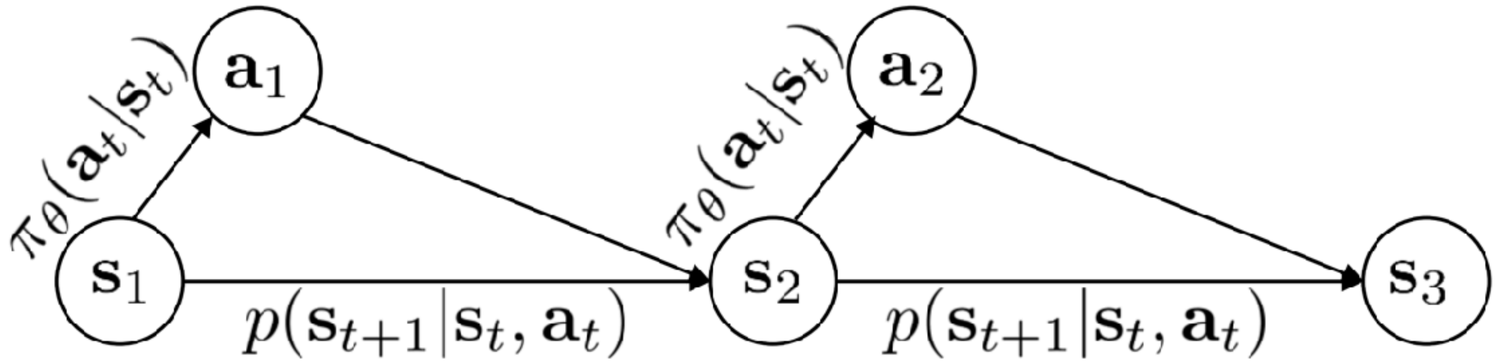
If you only have observations $\mathbf{o}$, give the policy memory: $\pi_\theta(\mathbf{a}_t\,|\,\mathbf{o}_{t-m}, \ldots, \mathbf{o}_t)$

What is the goal of reinforcement learning?

maximize sum of rewards: $\max \sum_t^T r(\mathbf{s}_t, \mathbf{a}_t)$

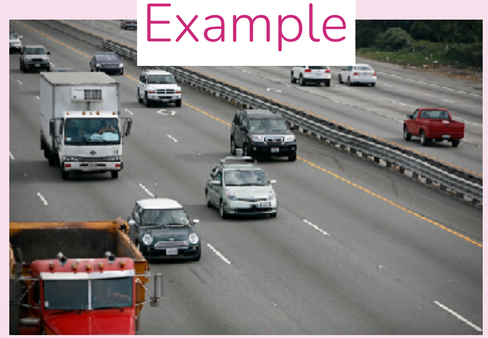
but this is not a deterministic quantity!

Question: what are the sources of variability?



$$\underbrace{p(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{p_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

Example



1. the world is stochastic
2. the car may not make the same decision every time.

What is the goal of reinforcement learning?

~~maximize sum of rewards: $\max \sum_t r(\mathbf{s}_t, \mathbf{a}_t)$~~

maximize *expected* sum of rewards: $\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$

$$\underbrace{p(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{p_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) p(\mathbf{s_{t+1}} \mid \mathbf{s}_t, \mathbf{a}_t)$$

Aside: why stochastic policies?

1. **Exploration**: to learn from your own experience, must try different things.

2. **Modeling stochastic behavior**: existing data will exhibit varying behaviors

We can leverage tools from **generative modeling**!

—> generative model over ***actions*** given states/observations

What is the goal of reinforcement learning?

maximize *expected* sum of rewards: $\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$

$$\underbrace{p(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{p_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) p(\mathbf{s_{t+1}} \mid \mathbf{s}_t, \mathbf{a}_t)$$

How good is a particular policy?

value function $V^\pi(\mathbf{s})$ - future expected reward starting at $\mathbf{s}$ and following π

Q-function $Q^\pi(\mathbf{s}, \mathbf{a})$ - future expected reward starting at $\mathbf{s}$, taking $\mathbf{a}$, then following π

Types of algorithms

maximize *expected* sum of rewards: $\max_{\theta} \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t^T r(\mathbf{s}_t, \mathbf{a}_t) \right]$

1. **Imitation learning**: mimic a policy that achieves high reward
2. **Policy gradients**: directly differentiate the above objective
3. **Actor-critic**: estimate value of the current policy and use it to make the policy better
4. **Value-based**: estimate value of the optimal policy
5. **Model-based**: learn to model the dynamics, and use it for planning or policy improvement

Why so many algorithms?

Algorithms make different trade-offs, thrive under different assumptions.

- How easy / cheap is it to collect data with policy? (e.g. simulator vs. hand-written)
- How easy / cheap are different forms of supervision? (demos, detailed rewards)
- How important is stability and ease-of-use?
- Action space dimensionality, continuous vs. discrete
- Is it easy to learn the dynamics model?

Let’s start with imitation learning.

- an important subroutine in some RL algorithms
- a powerful approach on its own

What is the goal of imitation learning?

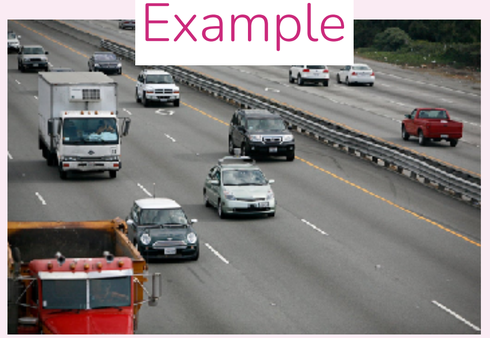
Data: Given trajectories collected by an expert

"demonstrations" $\mathscr{D} := \{(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T)\}$

(sampled from some unknown policy π_{expert})

Goal: Learn a policy π_θ that performs at the level of the expert policy, by mimicking it.

Example



- Dataset from human drivers
- Sensor readings + steering commands

Imitation learning - version 0

Deterministic policy

0. Given demonstrations collected by an expert $\mathscr{D} := \{(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T)\}$

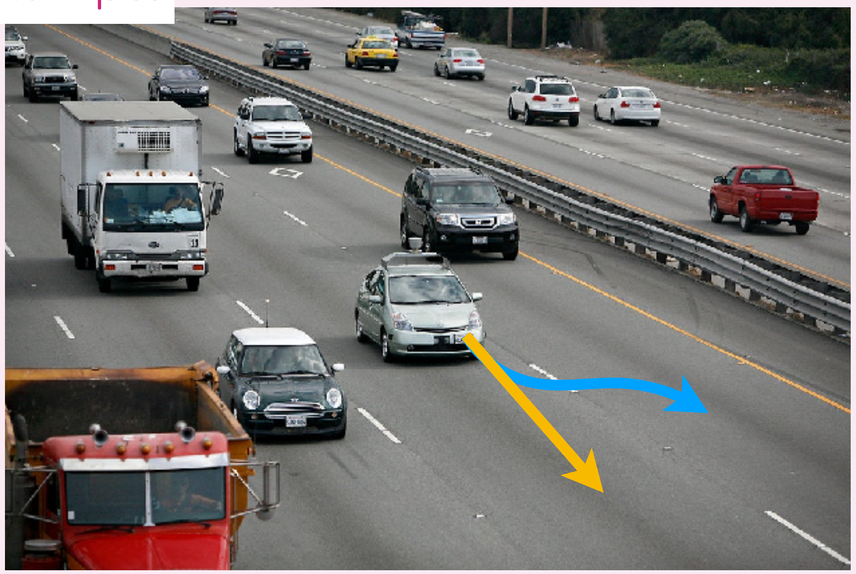
1. For deterministic policy, supervised regression to the expert's actions

$$\min_{\theta} \frac{1}{|\mathscr{D}|} \sum_{(\mathbf{s},\mathbf{a}) \in \mathscr{D}} \|\mathbf{a} - \hat{\mathbf{a}}\|^2 \qquad \text{where } \hat{\mathbf{a}} = \pi_\theta(\mathbf{s})$$

2. Deploy learned policy π_θ

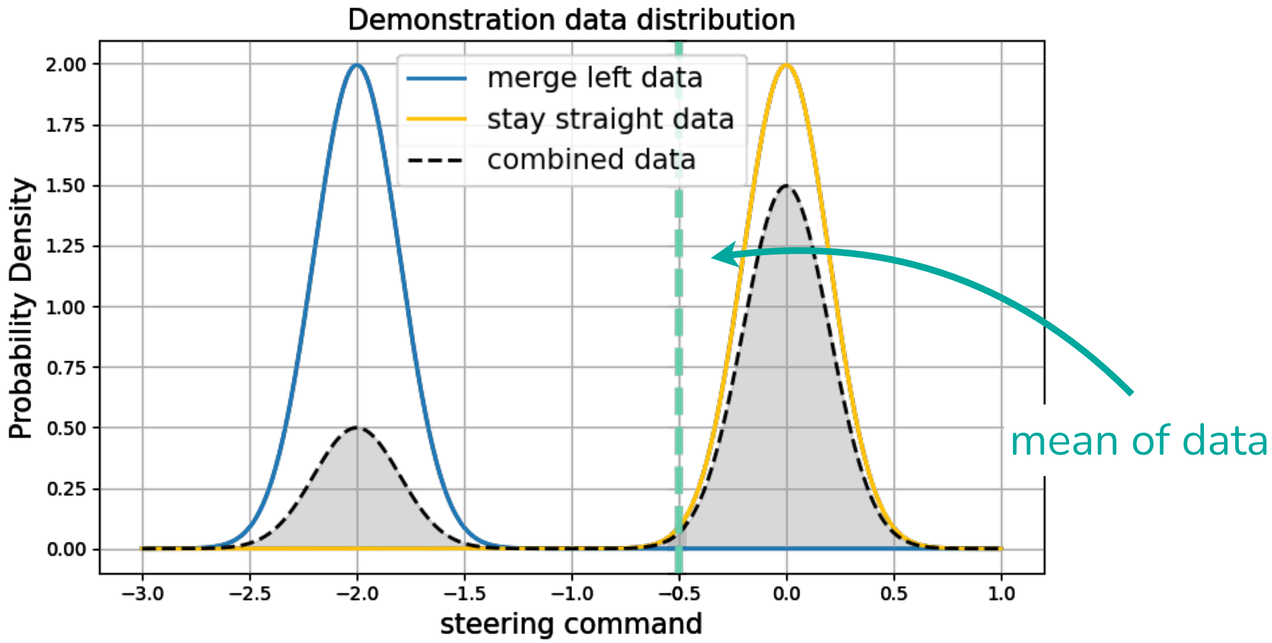
What could go wrong?

Example



- Dataset from human drivers
- Sensor readings + steering commands

Question: what might policy trained with ℓ_2-regression do?

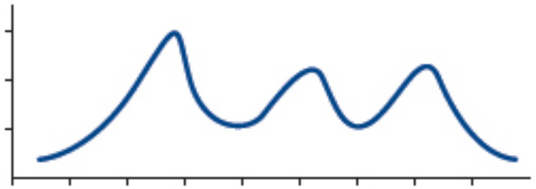


How can we represent more than the mean?

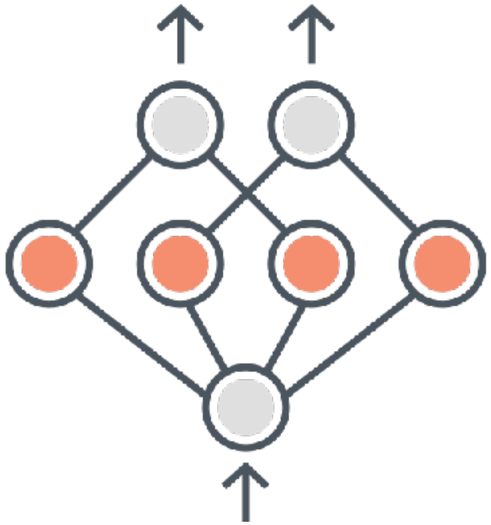
How often does this happen in practice? All the time!

Esp. when data collected by multiple people.

Learning *distributions* with neural networks

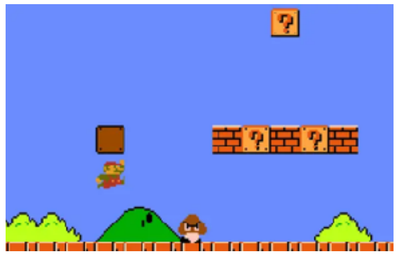


parameters of distribution

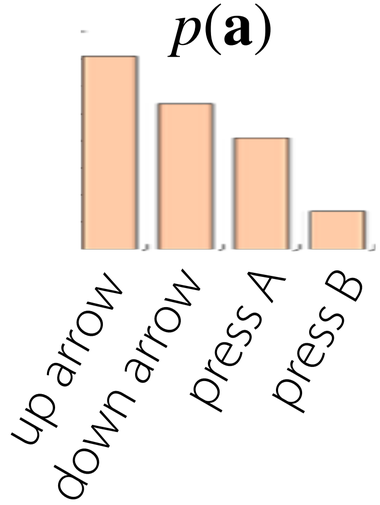


1D discrete actions

$\mathbf{s}$



$p(\mathbf{a})$



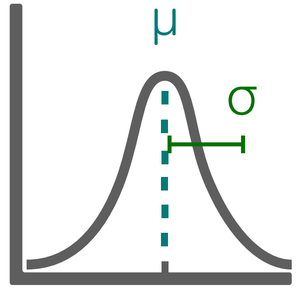
Neural net outputs $p(\text{up}), p(\text{down}), \ldots$ represent categorical distribution.

Maximally expressive

Continuous actions

$\mathbf{s}$

$p(\mathbf{a})$



next steering angle

Neural net outputs μ, σ to represent Gaussian distribution.

Not very expressive!

Next time: core imitation learning techniques, including more expressive distributions

Recap of definitions

state $\mathbf{s}_t$ - the state of the "world" at time t

(or *observation* $\mathbf{o}_t$ - what the agent observes at time t)

action $\mathbf{a}_t$ - the decision taken at time t

reward function $r(\mathbf{s}, \mathbf{a})$ - how good is $\mathbf{s}, \mathbf{a}$?

initial state distr. $p(\mathbf{s}_1)$, *unknown dynamics* $p(\mathbf{s_{t+1}} \mid \mathbf{s}_t, \mathbf{a}_t)$

(partially-observed) Markov decision process

MDP, POMDP

Recap of definitions

trajectory $\boldsymbol{\tau}$ - sequence of states/observations and actions $(\mathbf{s}_1, \mathbf{a}_1, \mathbf{s}_2, \mathbf{a}_2, \ldots, \mathbf{s}_T, \mathbf{a}_T)$

policy $\boldsymbol{\pi}$ - represents behavior, selecting actions based on states or observations

Goal: learn policy $\boldsymbol{\pi}_\theta$ that maximizes *expected* sum of rewards:

$$\max_\theta \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

value function $V^\pi(\mathbf{s})$ - future expected reward starting at $\mathbf{s}$ and following $\boldsymbol{\pi}$

Q-function $Q^\pi(\mathbf{s}, \mathbf{a})$ - future expected reward starting at $\mathbf{s}$, taking $\mathbf{a}$, then following $\boldsymbol{\pi}$

Course Reminders

Your Initial Steps:

Homework 1 due Fri 4/10 at 9 pm PT

Start forming final project groups if you want to work in a group

Coming Up Next:

Imitation Learning Lecture Cont. (Friday 9:30, NVIDIA Aud)

PyTorch Tutorial (Friday 3:30, Skilling Aud)