ORCS 4529
Reinforcement Learning

SHIPRA AGRAWAL

Course logistics

Instructor: Shipra Agrawal

- Class: Mon, Wed 1:10pm - 2:25pm
- Office hours: Tuesday 10:00am-11:00 am at Mudd 423

TA/CA

- TBD

Updated info on courseworks/canvas page

Communication: *Ed discussion*

Link on the left menu of courseworks

No emails (unless absolutely necessary)

Post questions on Ed discussion

- Can post publicly/privately/anonymously

Occasional zoom sessions

Regular classes will not be recorded

There may be makeup classes or recitations on zoom if needed

Link on the left menu of courseworks page

Any recordings will be under video library

Assignments: Gradescope

All assignments submission and grading through gradescope

Course requirements

Assignments

HW0 due Thursday Sep 18 midnight.

Lab 0 due Thursday Sep 18 midnight.

HW1 on MDP theory, will be released on Thursday September 18, due in two weeks after.

2 paper reading (with short report) assignments

4 additional lab assignments

Most deadlines will be Thursday midnight. Uniform and strict late submission policy.

Late submission policy

24 hour delay permitted

1 percentage point deducted for every 2 hours of delay (no deduction for less than two hours of delay)

Applies to all assignments and labs

Use of AI tools and academic conduct

- Any use of generative artificial intelligence (AI) tools such as ChatGPT for an assignment or lab must be appropriately acknowledged and cited prominently on top of the assignment. Each student is responsible for assessing the validity and applicability of any generative AI output that is submitted; you bear the final responsibility for the work you submit.
- In instances when collaborative work is assigned, the assignment should list all team members who participated.
- For individual assignments, students may discuss with others, but the solutions must be their own.
- Any use or copying of publicly available code or libraries must be must be appropriately acknowledged and cited.

Violations of these policies will be considered academic misconduct.

When in doubt – CITE and ACKNOWLEDGE!

Project

Most important component of the course.

Groups of 3-4

See canvas page for detailed

- Timeline
- Guidance on picking a topic and forming a group
- Writing a report and presentation

Overview of time line:

- **Mid-October:** pick topics and present in class, get feedback, and form groups of 3-4
- **Beginning of December:** Submit report and do presentations

Pre-requisites

Probability

Linear Algebra and Multivariate calculus

Machine learning

- Knowledge of Deep learning will be an advantage

Fundamentals of optimization algorithms

Coding in Python

- Experience with deep learning packages (tensorflow/pytorch etc.) will be an advantage

Course Introduction

What is Reinforcement Learning?

What do you think?

Reinforcement Learning

Science of sequential decision making

Agent interacts and learns from a stochastic environment

Many faces of reinforcement learning

- Dynamic Programming (Operations Research)
- Optimal control (Engineering)
- Reward systems (Neuro-science)
- Classical/Operant Conditioning (Psychology)

How is it different from Machine Learning?

Example

Imagine you have a mechanical programmable robot

Your goal is to train the robot to walk and go from one place to another.

Specific task: train the robot to walk from point A to point B

- Outcome: how to sequentially move limbs to accomplish this task

Propose some ways to do this using your ML toolbox.

What kind of data would you need? What ML techniques? What are the challenges?

Typical characteristics of a Reinforcement Learning problem

Need to make a sequence of decisions

No supervisor, only reward *signals*

- *Reinforce actions with good rewards*

Feedback is delayed

- *Credit assignment problem*

Decisions effect data (non i.i.d. training examples)

- *Need for exploration. Exploration-exploitation tradeoff.*

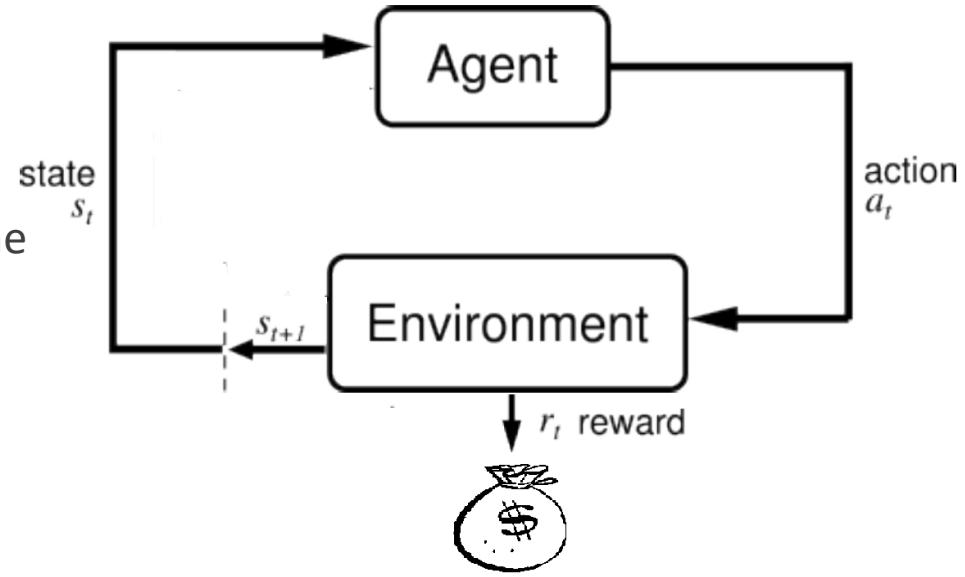
(Compare to your favorite supervised learning technique)

Modeling foundation: MDP

Sequential decisions
Reward signals
(partial labels)
Delayed feedback
Actions effect observations

Markov Decision process: a model for sequential decision making

- Past information is captured by **state**
- Agent takes an **action**, observes **reward and new state**
- **Markov Property:** New state and reward depend only on the current state and action
- Goal is to maximize is some long-term aggregate function of the immediate rewards



Reinforcement learning

Reinforcement learning ≡ MDP with unknown environment model

Agent observes samples : rewards, state transition

Aim is to find a good strategy (policy): *what action to take on observing each given state* in order to maximize aggregate reward

- implicitly or explicitly learn the environment model from observations
- optimize the objective

Examples

Automated vehicle control/robotics

- An unmanned helicopter learning to fly and perform stunts
- https://www.youtube.com/watch?v=M-QUkgk3HyE

Flying a helicopter

Actions/decisions

State

Environment

Reward

Goal

Examples

Automated vehicle control/robotics

- An unmanned helicopter learning to fly and perform stunts

Game playing

- Playing backgammon (Gerald Tesauro 1995), Atari games (Deepmind 2013), AlphaGo (Deepmind 2016)
- https://www.youtube.com/watch?v=V1eYniJ0Rnk
- https://gymnasium.farama.org/environments/atari/
- Landmark paper: Playing Atari using Deep Reinforcement Learning, Mnih et. al 2013


ima...
021 3

Playing a single player game

Actions/decisions

State

Environment

Reward

Goal

Examples

Automated vehicle control/robotics

- An unmanned helicopter learning to fly and perform stunts

Game playing

- Playing backgammon, Atari breakout, Tetris, Tic Tac Toe

Medical treatment planning

- Planning a sequence of treatments based on the effect of past treatments

Medical Treatment Planning

Actions/decisions

State

Environment

Reward

Goal

Examples

Automated vehicle control/robotics

- An unmanned helicopter learning to fly and perform stunts

Game playing

- Playing backgammon, Atari breakout, Tetris, Tic Tac Toe

Medical treatment planning

- Planning a sequence of treatments based on the effect of past treatments

Chat bots

- Agent figuring out how to make a conversation
- Dialogue generation, natural language processing
- (Chatgpt)

Course goals

Modeling a problem as RL/MDP

Rigorous understanding of the MDP foundation:

- Stochastic structure, algorithm design, convergence

Conceptual understanding of reinforcement learning algorithms

- Mathematical insights into design principles
- Some convergence results and theory on exploration-exploitation tradeoffs

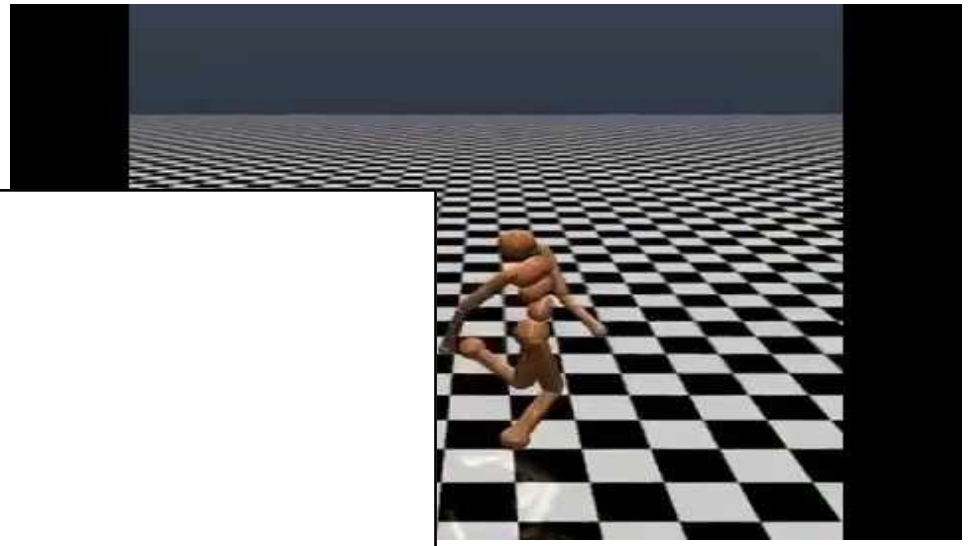
Ability to implement RL algorithms using some popular software platforms and simulators

- Utilize Deep learning with tensorflow or Pytorch
- [OpenAI] gym enviornments

Understanding connections to new and upcoming applications

Gymnasium (https://gymnasium.farama.org/)

Simulated environments for testing reinforcement learning algorithms



umanoid-v1

Breakout-v0

Topics

Introduction to MDP: value-iteration, policy iteration, Q-value-iteration

RL algorithm design

- **Q-learning and Deep Q-networks (DQN)**: Tabular, function approximation, deep network architecture, backpropagation, experience replay
- **Policy gradient methods**: Function approximation, Natural policy gradient, Trust region policy optimization, Actor critic methods

Exploration-Exploitation and multi-armed bandit theory

RLHF Reinforcement Learning with Human Feedback

MCTS Monte Carlo Tree Search and Connections to Agentic Systems

Multi-agent RL and connections to game theory

HW0 and Lab0

Checks for pre-requisites

Due in two weeks

TO DO

1. Check syllabus, project page on courseworks
2. HW0 and Lab0!
3. Start thinking about project! Use Ed discussion, use hallways

Basics of Deep Learning

Regression

Given labeled (training) data (x_i, y_i), fit function (model) f_θ so that $y_i = f_\theta(x_i)$

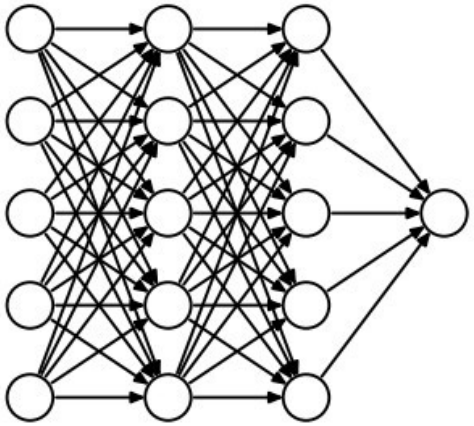
Linear regression

Function $f_\theta(x_i) = \theta^T x_i$ (linear model)

Fit θ to minimize loss

$$\frac{1}{\mathrm{n}}\sum_i (f_\theta(x_i) - y_i)^2 + R(\theta)$$

Convex optimization for linear functions f_θ



Deep Learning

Function $f_\theta(\cdot)$ is defined by a neural network

- Nonlinear Non-convex optimization

Basic architecture of neural network model f_θ

Multi-layer perceptron

Stack multiple layers of linear transformation & nonlinear functions

Input $x \in R^6$, first transform

$$W_1 x + b_1 \in R^4,$$

then apply nonlinear function (sigmoid/relu)

$$h_1 = \sigma(W_1 x + b_1).$$

Second layer transforms

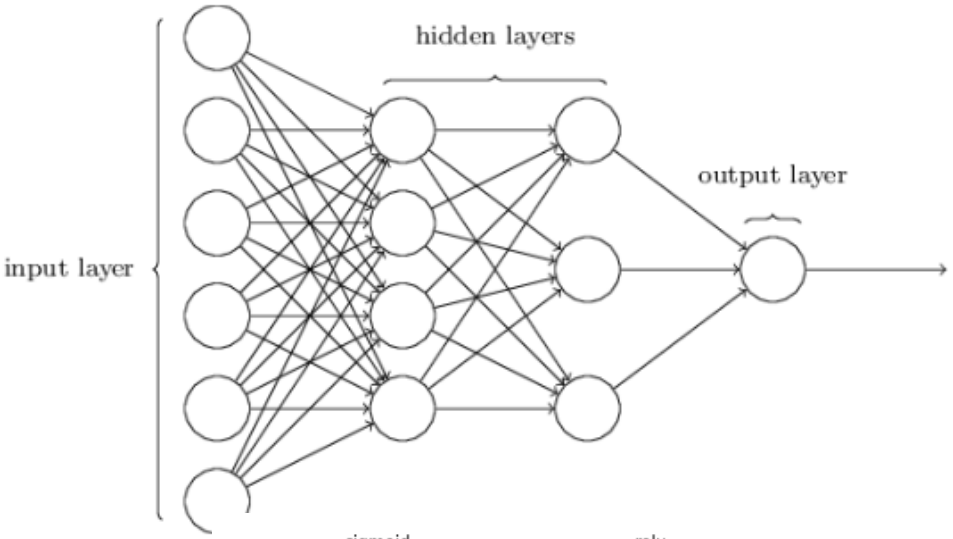
$$h_2 = \sigma(W_2 h_1 + b_2) \in R^3$$

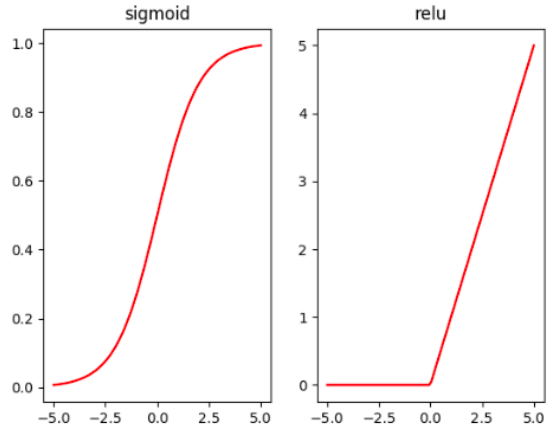
Final layer $\hat{y} = W_3 h_2 + b_3 \in R$

Overall

$$\hat{y}_i = W_3 \sigma(W_2 \sigma(W_1 x + b_1) + b_2) + b_3 =: f_\theta(x)$$

$$\theta = (W_1, b_1, W_2, b_2, W_3, b_3)$$





How do we train it?

Just like linear regression

Set least square loss function on training data

$$\frac{1}{n}\sum_{i}(f_\theta(x_i)-y_i)^2$$

Optimize for θ

- Nonlinear optimization techniques, but basically smart versions of gradient descent
- Backpropagation idea to make the gradient computation efficient

Lets get our hands dirty

https://colab.research.google.com

Classification

In training data (x_i, y_i),

- y_i are binary (0,1) labels
- or one of K classes $(a_1, a_2, \ldots, a_K)$

Predict probability of class of x_i to be y_i

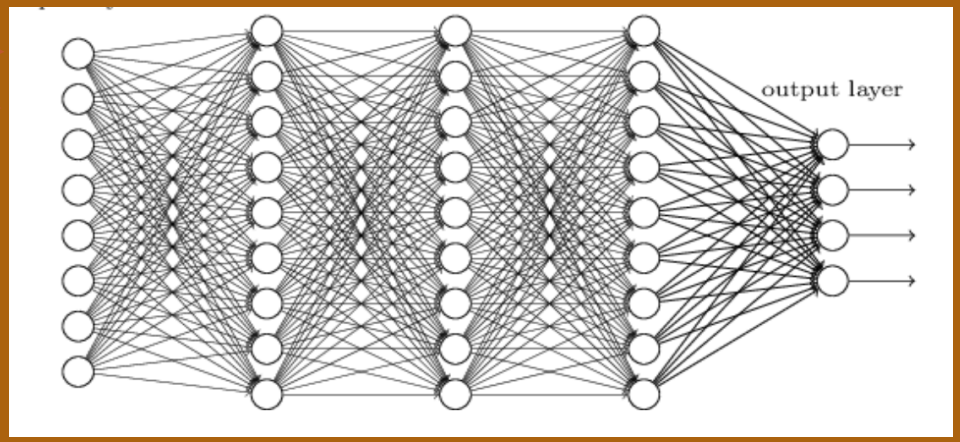
Fit a function $f_\theta(x, a)$ by maximizing likelihood of the training data

$$\text{Minimize Loss} = -\sum_i \; y_i \log p_i$$

$$\text{where } p_i = \frac{\exp(f_\theta(x_i, y_i))}{\sum_{j=1,\ldots,K} \exp(f_\theta(x_i, a_j))}$$

Classification using neural networks

$x \in R^8$



$f_\theta(x, a_1)$
$f_\theta(x, a_2)$
$f_\theta(x, a_2)$
$f_\theta(x, a_4)$

How is deep learning used in RL? Many ways.

Before taking any action, agent observes state of the environment.

Given historical data, supervised learning

- To predict how the state of the environment will change on taking a given action
- To predict immediate reward of a given action in a given state
- To predict long term objectives .. Details later

Lab 0

finish Lab0_regression and Lab0_classification either in torch or tensorflow

Additional slides

Imitation learning (High-level overview)

An expert (typically a human) provides a large set of demonstrations

- Typically only actions, no rewards

Learn the optimal policy by following, imitating the expert's decisions

- Imitate (Learn) instead of optimize
- Given state, predict the next action of the expert

Given very rich data, from many experts, we can also learn "new decision making policies" that combine multiple experts or optimize the reward model

Labeled (but non i.i.d. sequential) data – **supervised learning**

Inverse RL

An expert (typically a human) provides a large set of demonstrations

- Typically only actions, no rewards

Learn the reward model

- Assuming expert is following an optimal policy
- Inverse problem: what reward model is optimized by the given policy

Learn “new decision making policies” that optimize the reward model

Labeled (but non i.i.d. sequential) data – **supervised learning**

Imitation Learning /Inverse RL vs. Reinforcement Learning

Imitation Learning (aka apprenticeship learning) shines for tasks

at which humans are good at

very large amount of data from many experts is available

Very difficult to define immediate rewards

Reinforcement Learning shines for tasks

at which automated systems could potentially do it much better and/or in a completely different way – *superhuman performance*

Conducting online experiments is possible/not too inefficient

Reasonable immediate rewards can be defined

➢Can you think of some examples for each?