P9120 - Statistical Learning and Data Mining

Lecture 11 - Reinforcement Learning III

Min Qian

Department of Biostatistics, Columbia University

November 14th, 2024

Outline

1. MDP planning problem
   - Policy evaluation
   - Policy improvement
   - Policy iteration
   - Value iteration

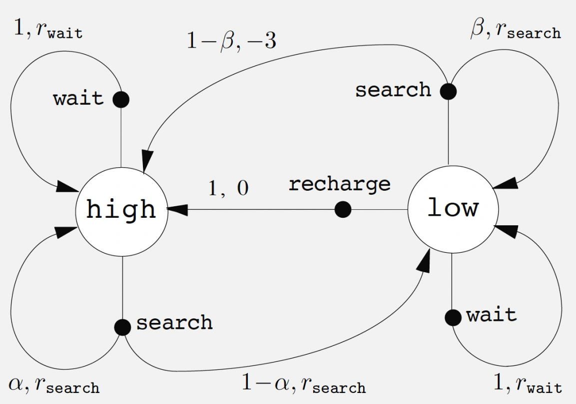
2. MDP learning problem
   - TD(0) for policy evaluation
   - Q-learning for estimating optimal policy

MDP Recap

$$S_0, A_0, R_0, \ldots, S_t, A_t, R_t, \ldots$$

- Markovian and stationary:
  system dynamics: $p(s', r|s, a) = \Pr(S_{t+1} = s', R_t = r|S_t = s, A_t = a)$
  state-transition: $p(s'|s, a) \triangleq \Pr(S_{t+1} = s'|S_t = s, A_t = a)$
  exp. rewards for (s, a, s'): $r(s, a, s') \triangleq E(R_t|S_t = s, A_t = a, S_{t+1} = s')$
  expected rewards for (s, a): $r(s, a) \triangleq E(R_t|S_t = s, A_t = a)$
- Return (cumulative rewards) $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k}$.
- Policy Deterministic policy $\pi(s) : \mathcal{S} \to \mathcal{A}$;
  Stochastic policy: $\pi(a|s) : \mathcal{S} \times \mathcal{A} \to [0, 1]$;
- (state-)value function of π: $V_\pi(s) = E_\pi(G_t|S_t = s)$
- Optimal policy $\pi^*(s) = \arg\max_\pi V_\pi(s)$.
  There always exists a deterministic optimal policy for an MDP.

Recycling Robot Example



- What's $V_\pi(s)$ of a given policy π?
  $\pi(\text{high}) = \text{search}$
  $\pi(\text{low}) = \text{wait}$

Bellman Equation for Value Function $V_\pi(s)$

Note that $G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k} = R_t + \gamma G_{t+1}$.

$$
\begin{aligned}
V_\pi(s) &\triangleq E_\pi[G_t | S_t = s] = E_\pi\left[R_t + \gamma G_{t+1} \middle| S_t = s\right] \\
&= E_\pi\left[R_t + \gamma E_\pi(G_{t+1} | S_{t+1}, S_t = s) \middle| S_t = s\right] \\
&= E_\pi\left[R_t + \gamma V_\pi(S_{t+1}) \middle| S_t = s\right]. \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r | s, a)[r + \gamma V_\pi(s')] \\
&= \sum_a \pi(a|s) \sum_{s'} p(s' | s, a)\left[r(s, a, s') + \gamma V_\pi(s')\right] \\
&= \sum_a \pi(a|s)\left[r(s, a) + \gamma \sum_{s'} p(s' | s, a) V_\pi(s')\right]
\end{aligned}
$$

$V_\pi(s)$ is the unique solution to a system of linear equations!

Recycling Robot Example

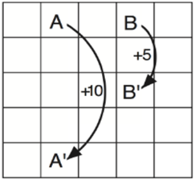
$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|s,a)\Big[r(s,a,s') + \gamma V_\pi(s')\Big]$$

s	a	s'	$p(s' \mid s,a)$	$r(s,a,s')$
high	search	high	α	$r_{\texttt{search}}$
high	search	low	$1-\alpha$	$r_{\texttt{search}}$
low	search	high	$1-\beta$	-3
low	search	low	β	$r_{\texttt{search}}$
high	wait	high	1	$r_{\texttt{wait}}$
high	wait	low	0	-
low	wait	high	0	-
low	wait	low	1	$r_{\texttt{wait}}$
low	recharge	high	1	0
low	recharge	low	0	-

What is $V_\pi(s)$ for policy $\pi(\text{high}) = \text{search}, \pi(\text{low}) = \text{wait}$?

Gridworld Example

$$V_\pi(s) = \sum_a \pi(a|s)\left[r(s,a) + \gamma \sum_{s'} p(s'|s,a)V_\pi(s')\right]$$



State space: 25 cells. Action space: ←, →, ↑, ↓

Dynamics of MDP (reward dist. and transition prob.)

$S_{t+1} = S_t, R_t = -1$ if A_t would take the agent off the grid

$S_{t+1} = A', R_t = +10$ if $S_t =$ A regardless of A_t

$S_{t+1} = B', R_t = +5$ if $S_t =$ B regardless of A_t

$S_{t+1} = S_t + \textit{one cell in the direction of } A_t,\ \ R_t = 0$, o.w.

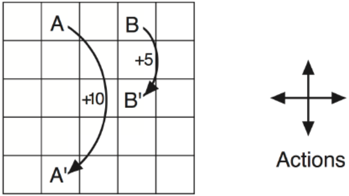
Question: What is the value function of a random policy (i.e., $\pi(a|s) = 1/4$ for all (s,a) pairs) with discount rate $\gamma = 0.9$?

Planning: Policy Evaluation

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$ arbitrarily, for $s \in \mathcal{S}$, and $V(\mathit{terminal})$ to 0

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$



3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

Policy Improvement: Value Functions

- (state-)Value function

$$V_\pi(s) \triangleq E_\pi(G_0 | S_0 = s) = E_\pi(G_t | S_t = s).$$

i.e., expected return if start in state s and then follow policy π.

- Action-value function (also called Q-function):

$$Q_\pi(s, a) \triangleq E_\pi(G_t | S_t = s, A_t = a).$$

expected return if start in s, take action a (once), then follow π.

- V_π can be computed from Q_π:

$$V_\pi(s) = E_\pi[Q_\pi(S_t, A_t) | S_t = s] = \sum_a \pi(a|s) Q_\pi(s, a).$$

- Bellman Equation for $Q_\pi(s, a)$

$$Q_\pi(s, a) = E_\pi\left[R_t + \gamma Q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a\right]$$

Policy Improvement Theorem

Let π and π' be a pair of policies such that, for all $s \in \mathcal{S}$,

$$Q_\pi(s, \pi'(s)) \geq V_\pi(s) \quad \textit{if } \pi' \textit{ is a deterministic policy,}$$

$$\textit{or} \quad \sum_a \pi'(a|s) Q_\pi(s, a) \geq V_\pi(s) \quad \textit{if } \pi' \textit{ is a stochastic policy.}$$

Then $V_{\pi'}(s) \geq V_\pi(s)$ for all $s \in \mathcal{S}$.

Proof.

Note that $Q_\pi(s, \pi'(s)) = E[R_t + \gamma V_\pi(S_{t+1}) | S_t = s, A_t = \pi'(s)]$

$$\begin{aligned}
V_\pi(s) \leq & E_{\pi'}[R_t + \gamma V_\pi(S_{t+1}) | S_t = s] \\
\leq & E_{\pi'}[R_t + \gamma E_{\pi'}[R_{t+1} + \gamma V_\pi(S_{t+2}) | S_{t+1}] | S_t = s] \\
= & E_{\pi'}[R_t + \gamma R_{t+1} + \gamma^2 V_\pi(S_{t+2}) | S_t = s] \\
\leq & E_{\pi'}[R_t + \gamma R_{t+1} + \gamma^2 E_{\pi'}[R_{t+2} + \gamma V_\pi(S_{t+3}) | S_{t+2}] | S_t = s] \\
\leq & E_{\pi'}[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \ldots | S_t = s] = V_{\pi'}(s)
\end{aligned}$$

$\square$

Planning: Policy Improvement

For a given policy π:

1. Compute Q_π from V_π:

$$\begin{aligned} Q_\pi(s,a) &= E_\pi\left[R_t + \gamma G_{t+1} | S_t = s, A_t = a\right] \\ &= E_\pi\left[R_t + \gamma V_\pi(S_{t+1}) | S_t = s, A_t = a\right] \\ &= \sum_{s',r} p(s',r|s,a)[r + \gamma V_\pi(s')] \\ &= r(s,a) + \gamma \sum_{s'} p(s'|s,a) V_\pi(s') \end{aligned}$$

2. Improve π by the greedy policy $\pi'(s) = \arg\max_a Q_\pi(s,a)$. Since

$$Q_\pi(s,\pi'(s)) = \max_a Q_\pi(s,a) \geq Q_\pi(s,\pi(s)) = V_\pi(s),$$

we have $V_{\pi'}(s) \geq V_\pi(s)$ for all s.

Policy Iteration for Computing Optimal Policy π^*

1. Initialization: arbitrary policy $\pi_0(s)$.
2. For $t = 0, 1, 2, \ldots$, repeat
    1. Policy evaluation: Initialize $V_{\pi_t}^{(0)}(s)$ for $s \in \mathcal{S}$. For $j = 0, 1, 2, \ldots$,

    $$V_{\pi_t}^{(j+1)}(s) \leftarrow \sum_{s' \in \mathcal{S}} p(s'|s, \pi_t(s)) \left[r(s, \pi_t(s)) + \gamma V_{\pi_t}^{(j)}(s') \right] \text{ for each } s \in \mathcal{S}$$

    until convergence. The converged value is denote as $V_{\pi_t}(s)$.
    2. If $\pi_t(s) = \arg\max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V_{\pi_t}(s') \right]$ for $\forall s \in \mathcal{S}$, stop and output $\pi^* = \pi_t$ and $V^* = V_{\pi_t}$.
    Otherwise, policy Improvement:

    $$\pi_{t+1}(s) \leftarrow \arg\max_a \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) V_{\pi_t}(s') \right].$$

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi^*$

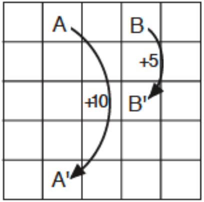
1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$; $V(terminal) \doteq 0$

2. Policy Evaluation
   Loop:
   $\quad \Delta \leftarrow 0$
   $\quad$ Loop for each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s))\left[r + \gamma V(s')\right]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$
   $\quad$ If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx V^*$ and $\pi \approx \pi^*$; else go to 2

Bellman Equation for Value (Gridworld Example)

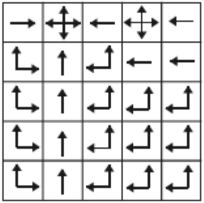
$$\pi_0 \xrightarrow{\text{policy evaluation}} V_{\pi_0} \xrightarrow{\text{policy improvement}} \pi_1 \xrightarrow{\text{policy evaluation}} V_{\pi_1} \xrightarrow{\text{policy improvement}} \pi_2 \xrightarrow{\text{policy evaluation}} V_{\pi_2} \xrightarrow{\text{policy improvement}} \cdots$$



Gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

V^*



π^*

Optimal Value Functions

- Optimal Policy $\pi^*(s) = \arg\max_\pi V_\pi(s)$.
- Optimal value function

$$V^*(s) \triangleq \max_\pi V_\pi(s) = V_{\pi^*}(s).$$

- Optimal action-value function (or optimal Q-function)

$$\begin{aligned} Q^*(s,a) &\triangleq \max_\pi Q_\pi(s,a) \\ &= \max_\pi E[R_t + \gamma V_\pi(S_{t+1}) | S_t = s, A_t = a] \\ &= E[R_t + \gamma V_{\pi^*}(S_{t+1}) | S_t = s, A_t = a] \\ &= Q_{\pi^*}(s,a) \end{aligned}$$

Bellman Optimality Equations

$$
\begin{aligned}
V^*(s) &= E_{\pi^*}[R_t + \gamma V_{\pi^*}(S_{t+1})|S_t = s] \\
&= Q_{\pi^*}(s, \pi^*(s)) = \max_a Q_{\pi^*}(s, a) \qquad (1) \\
&= \max_a E\Big[R_t + \gamma V^*(S_{t+1})\Big|S_t = s, A_t = a\Big] \\
&= \max_a \sum_{s'} p(s'|s, a)\Big[r(s, a) + \gamma V^*(s')\Big]
\end{aligned}
$$

$$
\begin{aligned}
\text{and } Q^*(s, a) &= E\Big[R_t + \gamma V^*(S_{t+1})\Big|S_t = s, A_t = a\Big] \\
&= E\Big[R_t + \gamma \max_a Q^*(S_{t+1}, a)\Big|S_t = s, A_t = a\Big] \\
&= \sum_{s'} p(s'|s, a)\Big[r(s, a) + \gamma \max_a Q^*(s', a)\Big]
\end{aligned}
$$

From (1), $\pi^*(s) = \arg\max_a Q^*(s, a)$.

Planning: Value Iteration

1. Find optimal value function using an iterative process based on bellman optimality equation.
2. Derive the optimal policy from the optimal value function

Value Iteration, for estimating $\pi \approx \pi^*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\mathit{terminal}) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $\quad v \leftarrow V(s)$
| $\quad V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$
| $\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi^*$, such that
$\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$

The Learning Problem

- The environment dynamic is unknown, but data are available.
- How to learn $V_\pi(\cdot)$, i.e., the value function for a given policy π?
  – Prediction problem
- How to learn the optimal policy π^*? – Control problem

Bellman Equations:

$$\begin{aligned} V_\pi(S_t) =& E_\pi\left[R_t + \gamma V_\pi(S_{t+1}) \middle| S_t\right] \\ Q^*(S_t, A_t) =& E\left[R_t + \gamma \max_a Q^*(S_{t+1}, a) \middle| S_t, A_t\right]. \end{aligned}$$

Basic Optimization

Data $(Y_1, \ldots, Y_n)$ with $Y_i = \mu + \epsilon_i$ and $E\epsilon_i = 0$ for $i = 1, \ldots, n$.

How to estimate μ?

- Analytically: $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} Y_i = \arg\min_{\mu} \frac{1}{2}\sum_{i=1}^{n}(Y_i - \mu)^2$.
- Gradient descent: $\hat{\mu}^{new} \leftarrow \hat{\mu}^{old} + \alpha \sum_{i=1}^{n}(Y_i - \hat{\mu}^{old})$.
- Stochastic gradient descent:
  update using one data point at a time $\hat{\mu}^{new} \leftarrow \hat{\mu}^{old} + \alpha(Y_i - \hat{\mu}^{old})$.

Bellman Equation: $V_\pi(S_t) = E_\pi\left[R_t + \gamma V_\pi(S_{t+1}) \middle| S_t\right]$

Generate data based on policy π:

$$R_t + \gamma V_\pi(S_{t+1}) = V_\pi(S_t) + \epsilon_t, \text{ where } E(\epsilon_t | S_t) = 0.$$

TD(0) for Prediction (i.e., Policy Evaluation)

Given $S_t \to$ choose $A_t = \pi(S_t) \to$ observe (S_{t+1}, R_t), update

$$\widehat{V}_\pi^{new}(S_t) \leftarrow \widehat{V}_\pi^{old}(S_t) + \alpha[R_t + \gamma V_\pi(S_{t+1}) - \widehat{V}_\pi^{old}(S_t)].$$

Replace $R_t + \gamma V_\pi(S_{t+1})$ by $R_t + \gamma \widehat{V}_\pi^{old}(S_{t+1})$.

Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated
Algorithm parameter: step size $\alpha \in (0, 1]$
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\textit{terminal}) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    $A \leftarrow$ action given by π for S
    Take action A, observe R, S'
    $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
    $S \leftarrow S'$
  until S is terminal

Q-Learning (Watkins, 1989)

Bellman Equation for optimal Q-function

$$Q^*(S_t, A_t) = E\left[R_t + \gamma \max_a Q^*(S_{t+1}, a) \middle| S_t, A_t\right]$$

implies that $R_t + \gamma \max_a Q^*(S_{t+1}, a) = Q^*(S_t, A_t) + \epsilon$, $E(\epsilon | S_t, A_t) = 0$.

- Off-policy: learn optimal policy while data collected from any policy.
- Useful if you cannot control the policy for data generation. (very useful in health application).
- Can reuse data generated from old policies.

Q-Learning (off-policy Control)

For any given (S_t, A_t, R_t, S_{t+1}):

$$\widehat{Q}^{new}(S_t, A_t) \leftarrow \widehat{Q}^{old}(S_t, A_t) + \alpha\left[R_t + \gamma \max_a Q^*(S_{t+1}, a) - \widehat{Q}^{old}(S_t, A_t)\right]$$

Replace $R_t + \gamma \max_a Q^*(S_{t+1}, a)$ by $R_t + \gamma \max_a \widehat{Q}^{old}(S_{t+1}, a)$.

Q-learning (off-policy TD control) for estimating $\pi \approx \pi^*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$
    $S \leftarrow S'$
  until S is terminal

Q-learning with Continuous State Space

- Bellman Equation $Q^*(S_t, A_t) = E\big[R_t + \gamma \max_a Q^*(S_{t+1}, a)\big| S_t, A_t\big]$.
- Approximate $Q^*(s, a)$ by $Q(s, a, ; \theta)$, and estimate θ by minimizing

$$\sum_t [R_t + \gamma \max_a Q(S_{t+1}, a; \theta) - Q(S_t, A_t, ; \theta)]^2.$$

- Stochastic gradient descent (viewing $Q(S_{t+1}, a; \theta)$ as a fixed target)

$$\widehat{\theta}^{new} \leftarrow \theta + \alpha\big[R_t + \gamma \max_a Q(S_{t+1}, a; \theta) - Q(S_t, A_t; \theta)\big] \frac{dQ(S_t, A_t; \theta)}{d\theta}\Big|_{\theta = \widehat{\theta}^{old}}.$$

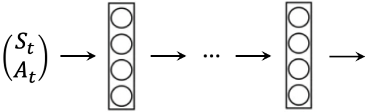
	Q-learning
Tabular	**Converges to q^* (Watkins and Dayan, 1992) (Tsitsiklis, 1994)**
Linear	**Can diverge (Wiering, 2004; Baird, 1995)**
Non-Linear	Can Diverge

Deep Q-Network (DQN)

Data tuples $\{(S_t, A_t, R_t, S_{t+1}), t = 0, 1, 2, \ldots\}$

Usually store $10,000$ most recent (S_t, A_t, R_t, S_{t+1}) in replay memory $\mathcal{D}$

1. Initialize $Q(s, a)$ (say, $Q(s, a) = 0$ for all (s, a)).
2. Randomly sample a mini-batch of tuples (S_t, A_t, R_t, S_{t+1}) from $\mathcal{D}$ and train the Q-function using neural network:

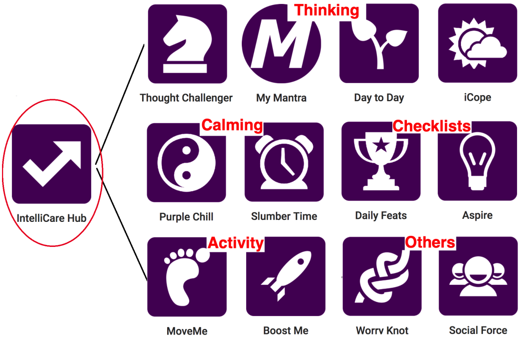


$$Y_t \triangleq \begin{cases} R_t & \text{if } S_{t+1} \text{ is terminal} \\ R_t + \gamma \max_a Q(S_{t+1}, a) & \text{else} \end{cases}$$

3. Set $Q = Q^{new}$

RL Mobile Health Applications

IntelliCare for Depression and Anxiety



The hub app:

- Recommend apps weekly
- Track app usage.
- Manage other apps and push notifications.

- What app(s) to recommend so as to maximize app usage?
  - app usage of current week
  - cumulative app usages over several weeks.
- When to send out push notification so the users will respond?

Reminders

- Final project proposal due at 9pm on December 2nd
  - Topic
  - The method you will investigate
  - Simulation, or real data analysis (what's the data set), or theoretical derivation of a novel method.

- Hw #3 is due at 9pm on November 16th

- In class group paper presentation (next week Nov 21st):
  - U-Net: Convolutional Networks for Biomedical Image Segmentation
    Team member: Manye Dong, Jiatong Li, Yiming Li, Wenxin Tian, Yuntian Xu, Shihang Zeng
  - Deep-learning-based real-time prediction of acute kidney injury outperforms human predictive performance.
    Team members: Ruoying Deng, Ruijie He, Ekaterina Hofrenning, Jessie Li, Authur Starodynov, Yueyi Xu