Robot Learning	**Assignment 1**
CS 4756 Spring 2024	**Due 11:59pm Sunday, February 18**

(1) Exploring Markov Decision Processes (5 points)

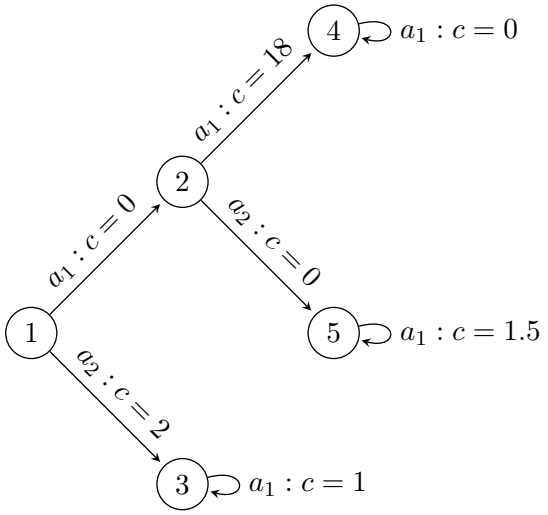


Figure 1: MDP for Problem 1

Compute the optimal value function V^* and the corresponding optimal policy π^* for each state in Fig. 1 for a discount factor of $\gamma = 0.9$ in the infinite horizon setting.

Notes:

- Initial State is always State 1.
- Each edge of the MDP is labeled in the following format: "{action} : {cost of action to complete transition}". Thus, the problem formulation involves a minimization of cost, rather than a maximization of a reward as may be seen elsewhere.
- Action a_1 at states 3, 4, and 5 must be taken infinitely if those states are ever reached.

Solution

We solve the Markov Decision Process (MDP) using the Bellman equation for cost minimization in an infinite horizon setting, given by:

$$V^*(s) = \min_a \left[c(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s') \right]$$

where $V^*(s)$ is the optimal value function for state s, $c(s,a)$ is the cost of taking action a in state s, γ is the discount factor, and $P(s'|s,a)$ is the transition probability from state s to state s' under action a.

Given the MDP structure from the problem, we calculate the optimal value functions:

For State 1:

$$V^*(1) = \min\{0 + 0.9 \cdot V^*(2), 2 + 0.9 \cdot V^*(3)\}$$

For State 2:

$$V^*(2) = \min\{18 + 0.9 \cdot V^*(4), 0 + 0.9 \cdot V^*(5)\}$$

For States 3, 4, and 5, where action a_1 is taken infinitely, we use the formula for the sum of an infinite geometric series:

$$V^*(3) = \frac{1}{1 - 0.9} = 10$$

$$V^*(4) = \frac{0}{1 - 0.9} = 0$$

$$V^*(5) = \frac{1.5}{1 - 0.9} = 15$$

Substituting back into the equations for States 1 and 2:

$$V^*(1) = \min\{0.9 \cdot V^*(2), 11\}$$

$$V^*(2) = \min\{18, 13.5\} = 13.5$$

Thus, the optimal value functions are:

$$V^*(1) = 11,\ V^*(2) = 13.5,\ V^*(3) = 10,\ V^*(4) = 0,\ V^*(5) = 15$$

The corresponding optimal policies π^* are:

- For State 1: Take action a_2 to go to State 3.
- For State 2: Take action a_2 to go to State 5.
- For State 3: Keep taking action a_1, the only action at State 3.
- For State 4: Keep taking action a_1, the only action at State 4.
- For State 5: Keep taking action a_1, the only action at State 5.

(2) A Story of Three Cliffs: Behavior Cloning and DAgger (10 points)

In this question, we are going to be thinking about robots falling off of cliffs and trying to get back on. We will look at three types of cliffs with varying levels of difficulty, and compare how well behavior cloning (BC) performs with respect to DAgger.
In all of the following parts, we will consider an infinite horizon setting with a discount factor of γ. Each part considers a Cliff MDP, where there exists a path atop the cliff consisting of "safe" states as well as a path to fall off the cliff from any point and land at the bottom, which we denote at s_x.

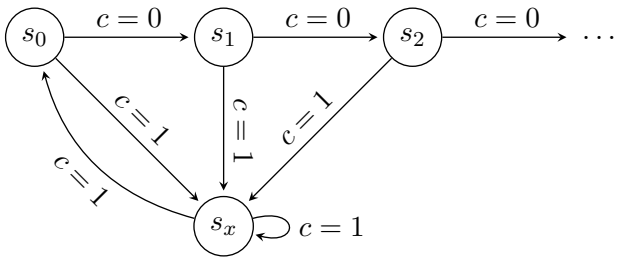
For each variant of the MDP, compute the tightest possible upper bounds (in terms of big-O) for the following quantities:

- $J(\pi_{BC})$: Expected total discounted sum of costs for Behavior Cloning
- $J(\pi_{DAgger})$: Expected total discounted sum of costs for DAgger

Important notes and assumptions:

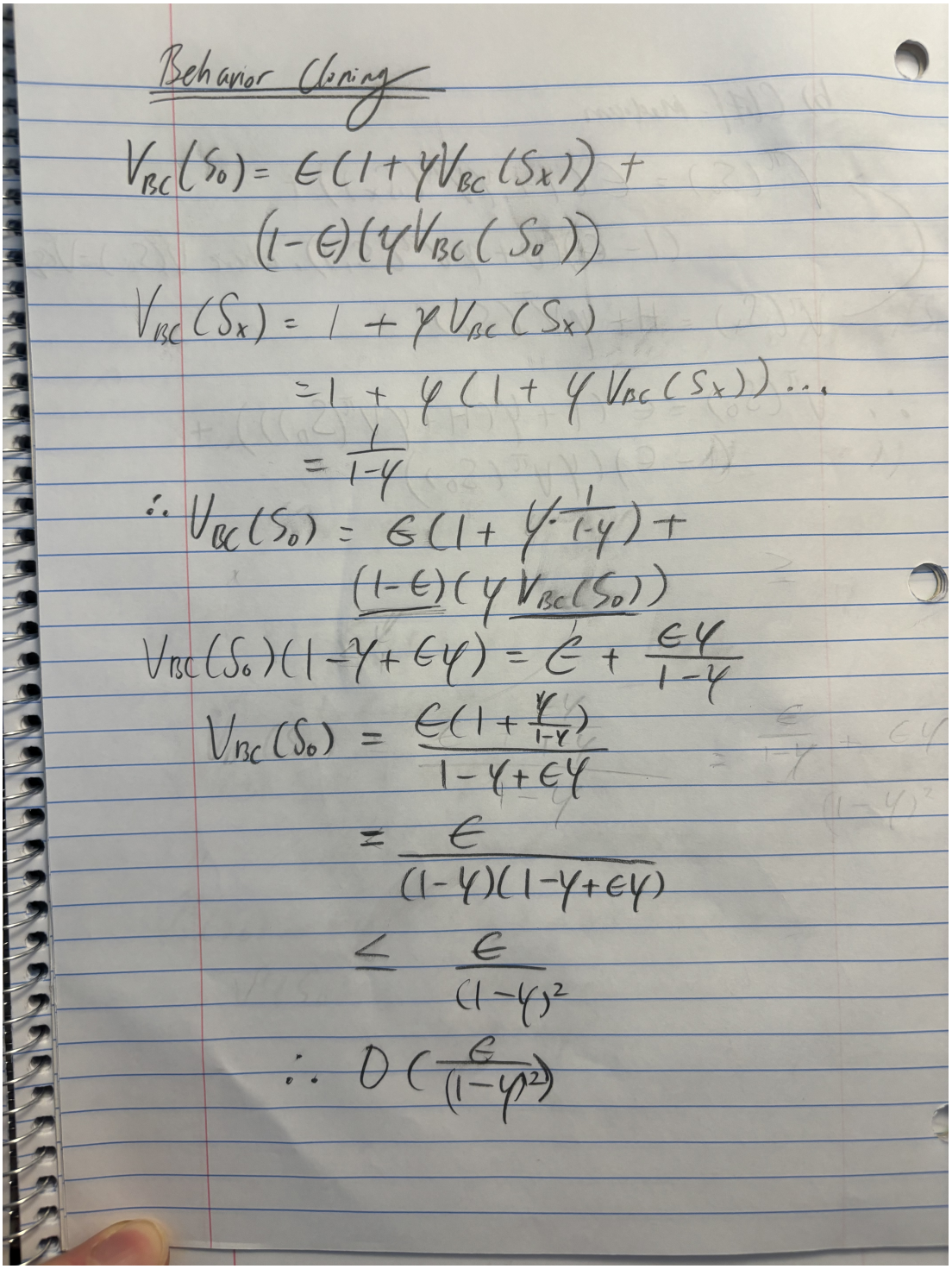
- An agent begins at state s_0
- The expert will always follow an optimal trajectory, thus the expert policy will incur 0 cost
- At each state the learner (both BC and DAgger) visits that the expert has also visited (i.e. all the safe states), it will make a mistake with probability ϵ and fall off the cliff
- Once the the BC learner has reached an unknown state, in the worst case with probability 1 it will continue to make mistakes and stay at the bottom of the cliff
- In contrast, the DAgger learner will query the expert to determine its next action, being able to complete a recovery action with probability 1 (if it exists)
- Assume that $0 < \epsilon \ll 1$, and $0 < 1 - \gamma \ll 1$ to simplify your calculations.

(a) Cliff-Easy



Cliff-Easy: The safe states s_i can either transition to s_{i+1} with $c = 0$ or s_x with $c = 1$, but there exists a recovery action from s_x to return to s_0 with $c = 1$. Bounds should be in terms of ϵ, γ.

Behavior Cloning



Behavior Cloning

$$V_{BC}(S_0) = \epsilon(1+\gamma V_{BC}(S_x)) + (1-\epsilon)(\gamma V_{BC}(S_0))$$

$$V_{BC}(S_x) = 1 + \gamma V_{BC}(S_x)$$

$$= 1 + \gamma(1 + \gamma V_{BC}(S_x)) \ldots$$

$$= \frac{1}{1-\gamma}$$

$$\therefore V_{BC}(S_0) = \epsilon\left(1 + \gamma \cdot \frac{1}{1-\gamma}\right) + (1-\epsilon)(\gamma V_{BC}(S_0))$$

$$V_{BC}(S_0)(1-\gamma+\epsilon\gamma) = \epsilon + \frac{\epsilon\gamma}{1-\gamma}$$

$$V_{BC}(S_0) = \frac{\epsilon\left(1+\frac{\gamma}{1-\gamma}\right)}{1-\gamma+\epsilon\gamma}$$

$$= \frac{\epsilon}{(1-\gamma)(1-\gamma+\epsilon\gamma)}$$

$$< \frac{\epsilon}{(1-\gamma)^2}$$

$$\therefore O\left(\frac{\epsilon}{(1-\gamma)^2}\right)$$

DAgger - Cliff-Easy

For the DAgger algorithm, the value at state s_0, denoted by $V_{DAgger}(s_0)$, is derived considering the fact that when the agent makes an error and transitions to state s_x, it queries an expert and takes a recovery action with probability 1, returning to s_0 with a cost of 1. Hence, we can write:

$$V_{DAgger}(s_0) = \epsilon \cdot (1 + \gamma \cdot V_{DAgger}(s_x)) + (1 - \epsilon) \cdot \gamma \cdot V_{DAgger}(s_0)$$
$$V_{DAgger}(s_x) = 1 + \gamma \cdot V_{DAgger}(s_0)$$

Substituting $V_{DAgger}(s_x)$ into the equation for $V_{DAgger}(s_0)$, we have:

$$\begin{aligned} V_{DAgger}(s_0) &= \epsilon \cdot (1 + \gamma \cdot (1 + \gamma \cdot V_{DAgger}(s_0))) + (1 - \epsilon) \cdot \gamma \cdot V_{DAgger}(s_0) \\ &= \epsilon + \epsilon\gamma + \epsilon\gamma^2 \cdot V_{DAgger}(s_0) + \gamma V_{DAgger}(s_0) - \epsilon\gamma V_{DAgger}(s_0) \end{aligned}$$

Isolating $V_{DAgger}(s_0)$, we get:

$$\begin{aligned} V_{DAgger}(s_0)(1 - \epsilon\gamma^2 - \gamma + \epsilon\gamma) &= \epsilon(1 + \gamma) \\ V_{DAgger}(s_0) &= \frac{\epsilon(1 + \gamma)}{1 - \gamma + \epsilon\gamma(1 - \gamma)} \end{aligned}$$

Assuming that ϵ is very small and γ is close to 1, the term $\epsilon\gamma(1-\gamma)$ is negligible. Therefore, we can approximate:

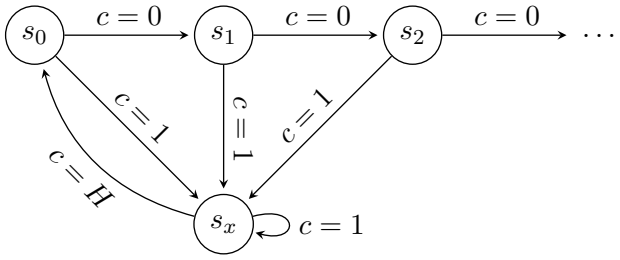
$$V_{DAgger}(s_0) \approx \frac{\epsilon(1 + \gamma)}{1 - \gamma}$$

Big O Notation for DAgger

Given the approximation and the assumptions, the tightest Big O bound for $V_{DAgger}(s_0)$, representing the expected total discounted sum of costs, is:

$$O\left(\frac{\epsilon}{1 - \gamma}\right)$$

(b) Cliff-Medium



Cliff-Medium: The safe states s_i can either transition to s_{i+1} with $c = 0$ or s_x with $c = 1$, but there exists a recovery action from s_x to return to s_0 with $c = H$. Bounds should be in terms of ϵ, γ, H.

DAgger - Cliff-Medium

For the DAgger algorithm, the value at state s_0, denoted by $V_{DAgger}(s_0)$, is derived considering the fact that when the agent makes an error and transitions to state s_x, it queries an expert and takes a recovery action with probability 1, returning to s_0 with a cost of H. Hence, we can write:

$$V_{DAgger}(s_0) = \epsilon \cdot (1 + \gamma \cdot V_{DAgger}(s_x)) + (1 - \epsilon) \cdot \gamma \cdot V_{DAgger}(s_0)$$
$$V_{DAgger}(s_x) = H + \gamma \cdot V_{DAgger}(s_0)$$

Substituting $V_{DAgger}(s_x)$ into the equation for $V_{DAgger}(s_0)$, we have:

$$\begin{aligned} V_{DAgger}(s_0) &= \epsilon \cdot (1 + \gamma \cdot (H + \gamma \cdot V_{DAgger}(s_0))) + (1 - \epsilon) \cdot \gamma \cdot V_{DAgger}(s_0) \\ &= \epsilon + \epsilon\gamma H + \epsilon\gamma^2 \cdot V_{DAgger}(s_0) + \gamma V_{DAgger}(s_0) - \epsilon\gamma V_{DAgger}(s_0) \end{aligned}$$

Isolating $V_{DAgger}(s_0)$, we get:

$$\begin{aligned} V_{DAgger}(s_0)(1 - \epsilon\gamma^2 - \gamma + \epsilon\gamma) &= \epsilon(1 + \gamma H) \\ V_{DAgger}(s_0) &= \frac{\epsilon(1 + \gamma H)}{1 - \gamma + \epsilon\gamma(1 - \gamma)} \end{aligned}$$

Assuming that ϵ is very small and γ is close to 1, the term $\epsilon\gamma(1 - \gamma)$ is negligible. Therefore, we can approximate:

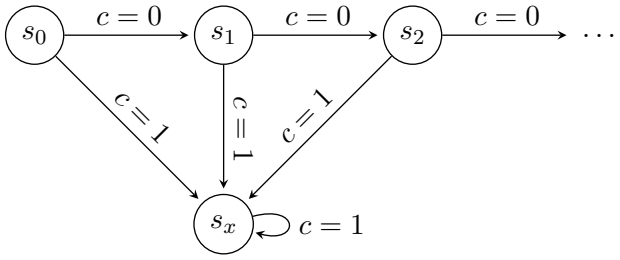
$$V_{DAgger}(s_0) \approx \frac{\epsilon(1 + \gamma H)}{1 - \gamma}$$

Big O Notation for DAgger

Given the approximation and the assumptions, the tightest Big O bound for $V_{DAgger}(s_0)$, representing the expected total discounted sum of costs, is:

$$O\left(\frac{\epsilon H}{1 - \gamma}\right)$$

(c) Cliff-Hard



Cliff-Hard: The safe states s_i can either transition to s_{i+1} with $c = 0$ or s_x with $c = 1$, but there is NO recovery action at s_x. Bounds should be in terms of ϵ, γ.

Cliff-Hard Analysis

We again separate the analysis for the cost at state s_0, the cost at state s_1 (which is the same as state s_0), and the cost at the failure state s_x.

First, let's analyze the cost at the failure state s_x. As there is no recovery action in Cliff-Hard, the cost of being in s_x is an infinite sum of costs incurred due to staying in s_x indefinitely. This sum can be represented as:

$$\begin{aligned} V(s_x) &= 1 + \gamma + \gamma^2 + \gamma^3 + \ldots \\ &= \frac{1}{1-\gamma} \end{aligned}$$

Now, considering the states s_0 and s_1, the value functions are as follows:

$$V(s_0) = (1-\epsilon) \cdot \gamma \cdot V(s_1) + \epsilon \cdot V(s_x)$$

Now, note that this looks awfully similar to Behavior Cloning, please refer back to the Behavior Cloning portion for the rest of the derivation of the upper bound.

Given that $V(s_1) = V(s_0)$ due to the symmetry of the states, we solve for $V(s_0)$:

$$V(s_0) = \frac{\epsilon}{(1-\gamma)^2}$$

Thus, the Big O bound for $V(s_0)$, representing the expected total discounted sum of costs, is:

$$O\left(\frac{\epsilon}{(1-\gamma)^2}\right)$$

Finally, comment on the following: How does the gap between DAgger and BC change as we vary the H in cliff-medium from 1 to $\frac{1}{1-\gamma}$? What is the explanation for this trend?

Comment on the Gap between DAgger and BC

As the recovery cost H in the Cliff-Medium scenario varies from 1 to $\frac{1}{1-\gamma}$, the performance gap between DAgger and Behavioral Cloning (BC) will also change.

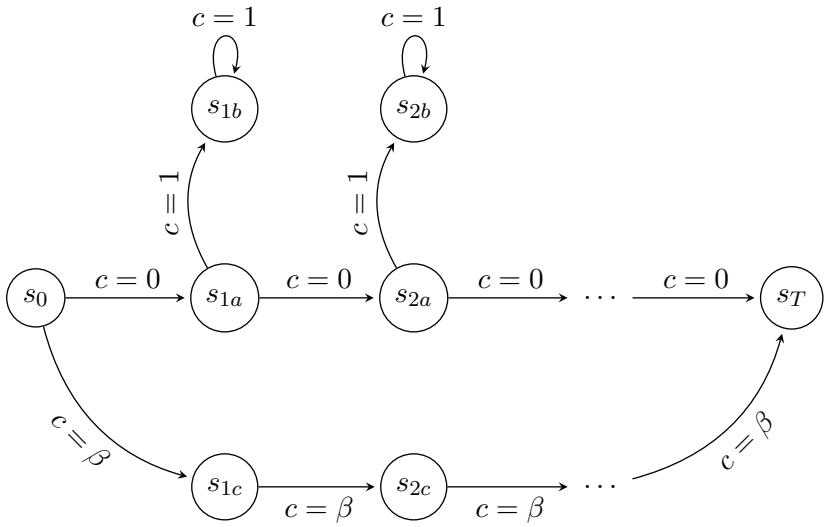
1. When $H = 1$, DAgger incurs a fixed cost for recovery, whereas BC may face repetitive costs due to not learning the correct policy.

2. As H increases towards $\frac{1}{1-\gamma}$, the cost of recovery in DAgger becomes significantly higher. This increases the total cost for DAgger, reducing its advantage over BC.

3. At $H = \frac{1}{1-\gamma}$, the recovery cost in DAgger becomes extremely high. This will make the big O cost for DAgger the same as the cost for BC, since they now have the same bound. s

The explanation for this trend lies in the increasing penalty for mistakes under DAgger. As H increases, the cost of each error made by the DAgger agent increases, potentially making it less efficient than BC, which does not incur additional costs for recovery actions.

(3) [Extra Credit] Pitfalls of DAgger (5 points)
In this problem, we explore the performance bound of a policy learned by DAgger in an interesting MDP with multiple routes to a final goal state.



Two ways to the goal

The MDP has two routes from the start s_0 to the goal s_T:

1. One route is the optimal route with action cost 0 that the expert takes. There is, however, a caveat. The route passes over a bridge $(s_{1a}, s_{2a}, \dots,)$ where a mistake can result in falling off the path into a ditch $(s_{1b}, s_{2b}, \dots)$. Once in the ditch, there is no recovery.
2. One route is a long detour with action cost β.

Assume the following about our learner

- On states on the bridge, $(s_{1a}, s_{2a}, \dots,)$, the learner makes a mistake with probability ϵ.
- On all other states, it can perfectly imitate the expert.

Compute the upper bound on $J(\pi_{DAgger})$, the expected total cost of trajectories for a DAgger policy over T timesteps. Now, compute the cost of the best policy in the learner's class, $J(\pi_L^\star)$.

- What is surprising about DAgger's performance specifically from this MDP based on this bound? Does DAgger find the best policy in the learner's policy class?
- In a few sentences, can you describe a real-world example that has a similar characteristic as this MDP?

Pitfalls of DAgger

The expected cost for the optimal policy is to take the detour, which incurs a total cost of $\beta * T$. However, because DAgger tries to mimic the expert, and the expert will correct the action before the detour is ever taken so that the dataset that is seen by the learner will not have the detour data in it. Therefore the cost for DAgger for this is similar to Cliff-Hard or Behavior Cloning, where there is no recovery action available for the learner. The upper bound on cost will be

$$O\left(\frac{\epsilon}{(1-\gamma)^2}\right)$$

Or

$$O(\epsilon * T^2)$$

$$\begin{aligned}
J(\lambda) &= \sum_{t=t_0}^{T-1} \mathbb{E}_{\xi_t \sim d^\lambda} c(\xi_t, \lambda(\xi_t)) \\
&= \epsilon \times (1+1+1+\cdots) \\
&\quad + (1-\epsilon) \times (0 + \epsilon \times (1+1+1+\cdots))_{T-1} \\
&\quad + (1-\epsilon) \times (0_t \cdots) \\
&= \epsilon T + (1-\epsilon)\epsilon(T-1) + (1-\epsilon)^2\epsilon(T-2) + \cdots \\
&= \epsilon \left(T + (1-\epsilon)(T-1) + (1-\epsilon)^2(T-2) + \cdots\right) \\
&\leq \epsilon \left(T + (T-1) + (T-2) + \cdots + 3 + 2 + 1\right) \\
&\leq \epsilon \cdot \frac{T(T+1)}{2} \approx O\left(\frac{\epsilon T^2}{2}\right)
\end{aligned}$$

What is surprising?

The surprising aspect might be that despite the DAgger policy's aim to mimic the expert, it could end up performing worse than a simple detour policy. This is primarily because the learner's mistake probability on the bridge makes the optimal expert route risky.

Does DAgger find the best policy?

It most likely does not in this case. Depending on the values of ϵ, β, and T, DAgger may or may not find the best policy in the learner's policy class. If the risk of falling into the ditch outweighs the cost of taking the detour, the DAgger policy may not be the best.

Real life example

Real-World Example: A real-world analogy could be self-driving cars learning to navigate in complex urban environments. An expert (human driver) might take more direct routes that are risky for an autonomous vehicle due to uncertainties in perception or decision-making. Where the best learner for this self driving car might choose longer, simpler routes to ensure safety, akin to taking the detour in the MDP, DAgger might end up trying to imitate the expert so it never sees the data for the detour, and ends up performing worse.

Attributions

Minimum use of Generative AI was adopted to format LaTeX and reword. The ideas and calculations are purely individual.