Reinforcement Learning

start time:

Before you start, share this document with your team member(s) and then complete the form
below to assign the role of speaker.

Team Role	Team Member
Speaker: shares your team’s ideas with the class.	Addy, Arogya,

A. Reinforcement Learning	start time:

Reinforcement Learning (RL) is a type of machine learning where an agent learns how to make decisions by interacting with an environment. The agent performs actions and learns from the feedback (rewards or penalties) it receives. The goal is for the agent to learn the best actions to take in order to maximize its total reward over time.

1. In MDP from the previous two activities, what information are required to know in order to learn the optimal policy?

    - State of the environment
    - List of possible actions for each state
    - Rewards and Transitions(probability)
    - Position of Terminals
    - Discount rate

2. Can we still use value or policy iteration to learn the optimal policy without knowing the transition model and rewards?

    - No we need to know the probability and rewards.

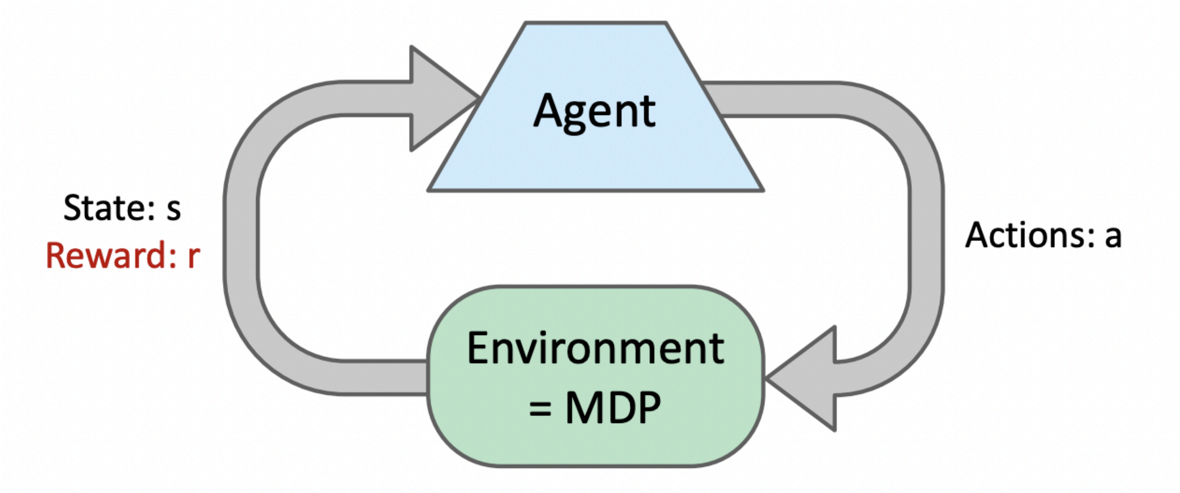
3. If you answer in 2 is no, we will need to develop new algorithms, reinforcement learning, to learn policy when we do not know the full information about the environment. List a couple of examples here like a game, project, or experiences from real world which we need to use RL to train a policy.

    - Traveling a maze
    - Predictive modeling and generative AI
    - ESCAPE ROOM

4. Is there another reason that RL is more pratically better then value/policy iteration even we know the full information of the environment？

    - We will not have to iterate over all the states in RL.

B. Model based Reinforcement Learning	start time:

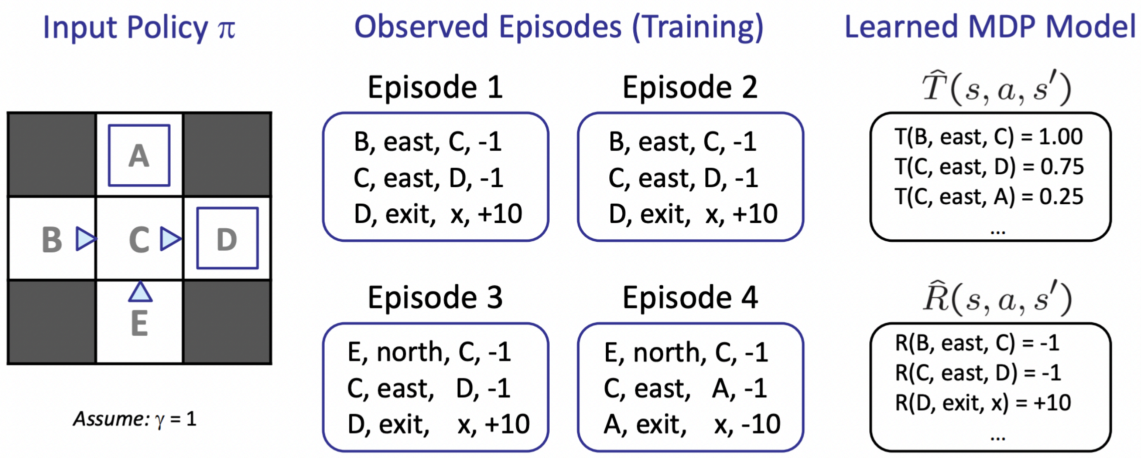


If we do not know the reward function *R(s, a, s')* or the transition functions *T(s, a, s')*, then we might want to learn R and T from the interactions with the environment.

Agent takes an action in the environment, the environment will then return new state and reward for that sate. This is called model based reinforcement learning.

5. We call this method model-based RL. Why? (We call RL model based if we use bellman equation)
    a. explicitly learning from the environment(model) to continually update the current state and select the next best action
    b. because we have a model that takes in action and provides us reward and transition values and we use those values to determine the action.

6. We will use **passive** reinforcement learning to identify functions R and T because while these functions are unknown, the policy being evaluated is **fixed**.



Input Policy π

Assume: $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Learned MDP Model

$\widehat{T}(s, a, s')$

T(B, east, C) = 1.00
T(C, east, D) = 0.75
T(C, east, A) = 0.25
...

$\widehat{R}(s, a, s')$

R(B, east, C) = -1
R(C, east, D) = -1
R(D, exit, x) = +10
...

Given a policy pi, we run a couple of episodes of the game and record the rewards and transitions. After running, we use the records to estimate the R and T. Now please do your own calculation to verify the learned T and R in the above example.

T(B, east, C) = 2/2 = 1 (no out of 4 because Episode 3 and 4 starts at E nto B and follows direction of arrows)

T(C, east, D) = ¾ = 0.75

T(C, east, A) = ¼ = 0.25

R(B, east, C) = (-1-1)/2 = -1

R(C, east, D) = (-1-1-1)/3 = -1

R(D, exit, x) = (+10+10+10)/3 = +10

7. Write a high level framework of this model-based RL. The input is a MDP environment without knowing T and R. The output is a policy.

input = environment

output = policy

take the environment with current state.

play game and extract state and reward

value iteration:

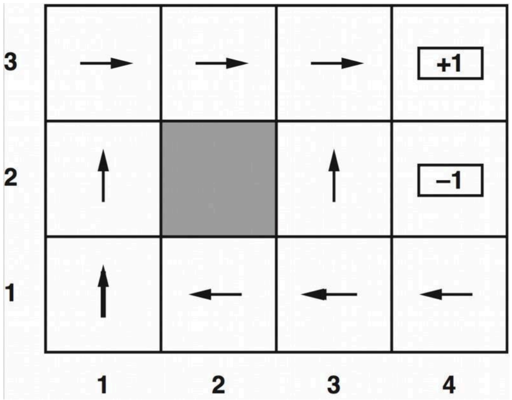
- for each state, take all possible actions and find average reward(extract R & T)
- best reward is kept in policy dictionary
- repeat for all states until we find all policies
- update policy in respect to best value V(pi)
- repeat everything until policy found and return

test the policy with environment.

C. Direct Utility Estimation - A Model Free Reinforcement Learning

start time:

Assume in Grid World, we have a fixed policy to follow but we do not know the R and T. For example, the policy being evaluated might be:



In passive reinforcement learning, our agent will execute a set of trials using the policy and see what happens (what **utility** is earned). In each trial, the agent starts at (1,1) until it reaches one of the terminal states, where we are trying to calculate the *$U^{\pi}(s)$* is the utility being estimated for a state s, with a discount factor γ=1 and the additive reward for non-terminal transitions is –0.04.

Assume the first trial is:

$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (2,3)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (4,3)_{\mathbf{+1}}$$

8. The numbering for the states differs from the numbering we've used.
    a. Where is the start state?
        i. (1,1)

    b. Where are the two terminal states?
        i. (4,3) & (4,2)

    c. In the first trial, which terminal state was reached?
        i. (4,3)

    d. Why does the state (1,2) appear twice in the first trial?
        i. .10 probability to travel East

9. To estimate the utility state, we work backwards from the terminal state that was reached in the trial. Because the (3,3) led to (4,3), where the reward was +1, the utility of state (3,3), its utility is 0.96.

    e. Why is the utility value is 0.96 and not 1?

        i. constant living penalty of -0.04 (penalty(-0.04) + reward(1))

    f. Fill in the other utility values for the first trial. If a state is reached multiple times, **write both values in**.

(1-(0.04*5)) = 0.8, (1-(0.04*3)) = 0.88	(1-(0.04*2)) = 0.92	(1-(0.04*1)) = 0.96	**+1**
(1-(0.04*6)) = 0.76, (1-(0.04*4)) = 0.84	**X**		**-1**
1-(0.04*7) = 0.72			

10. Now assume the agent has executed the policy a second time, with the following trial results. Carry out the series of state transitions and back-propagate the resulting cell values.

$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (1,2)_{\mathbf{-.04}} \rightsquigarrow (1,3)_{\mathbf{-.04}} \rightsquigarrow (2,3)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (3,2)_{\mathbf{-.04}} \rightsquigarrow (3,3)_{\mathbf{-.04}} \rightsquigarrow (4,3)_{\mathbf{+1}}$$

(1-(0.04*5)) = 0.80	(1-(0.04*4)) = 0.84	(1-(0.04*3)) = 0.88, (1-(0.04*1)) = 0.96	**+1**
(1-(0.04*6)) = 0.76	**X**	(1-(0.04*2)) = 0.92	**-1**
1-(0.04*7) = 0.72			

11. Here is a third trial. Carry out the series of state transitions and back-propagate the resulting cell values.

$$(1,1)_{\mathbf{-.04}} \rightsquigarrow (2,1)_{\mathbf{-.04}} \rightsquigarrow (3,1)_{\mathbf{-.04}} \rightsquigarrow (3,2)_{\mathbf{-.04}} \rightsquigarrow (4,2)_{\mathbf{-1}} .$$

			+1
	X	-1-(0.04*1)= -1.04	**-1**
-1-(0.04*4) = -1.16	-1-(0.04*3) = -1.12	-1-(0.04*2)= -1.08	

12. Based on these three trials, average the values discovered for each state and fill in below. If you have no values for a state, you can leave the value blank.

(.80+ .80+ .88) /3 = .83	(.92 + .84) / 2 = .88	(.96 + .88 + .96) / 3 = .93	**+1**

(.76 + .76 + .84) /3 = .786	**X**	(0.92-1.04)/2 = -0.06	**-1**
(.72 + .72 + (-1.16))/3 = .093	-1.12	-1.08	

13. We can continue to run more trials to update the estimation. How many trials do we need in this method?

14. **Direct Utility Estimation is** the name of this learning method in this section. Is this method a passive RL?
Yes it is passive because it strictly follows the policy.

15. Why do we call this method model free?
The agency doesn't want to learn the transition of model. Instead, this method estimates the policy of utility.

16. Can we call the idea of method Monte Carlo? If you do not know Monte Carlo method, let's check out this method to estimate the value of pi:
https://en.wikipedia.org/wiki/File:Pi_monte_carlo_all.gif

Yes because there is some sort of randomness in the

17. Do we learn any new policy yet from this method?
No, we just know the rewards at each states.

18. What is the disadvantage of using this method in estimating the utilities?
a. Inability to generalize, increasing error factor
b. We did not use Bellman style equation
c. It takes a lot of time to go through all of the policies

D. Temporal Difference (TD) Reinforcement Learning	start time:

Each **sample** of s is the cumulative reward from s until the environment is terminated in a single trial. One trial may contain multiple samples of s if s is visited more than one time in this trial. In the previous section, we simply take the average of all samples of s as the estimated utility value at s.

At each time step, an agent takes an action π(s) from a state s, transitions to a state s′, and receives a reward R(s,π(s),s′). We can obtain a sample value by summing the received reward with the discounted current value of s′ under π:

sample = R(s,π(s),s′)+γ$V^{\pi}(s')$

In temporal difference reinforcement learning, instead of weighting all samples equally like in direct utility estimation from previous section, we can weight newer samples higher, as in:

$$V^{\pi}(s) = (1 - \alpha)V^{\pi}(s) + (\alpha)\, sample$$

V(s) is the current value and sample is the newly estimated value. We use learning rate alpha to balance the weighted average of current and new values together as the next estimated value at s.

Remark:

$$\begin{aligned} V^{\pi}(s) &= (1 - \alpha)V^{\pi}(s) + (\alpha)\, sample \\ &= (1 - \alpha)V^{\pi}(s) + (\alpha)\,(R(s, s') + \gamma * V^{\pi}(s')\ ) \\ &= V^{\pi}(s) + \alpha * (R(s, s') + \gamma * V^{\pi}(s') - V^{\pi}(s)\ ) \end{aligned}$$

So, we use $R(s, s') + \gamma * V^{\pi}(s') - V^{\pi}(s)$ to update the estimated utility $V^{\pi}(s)$ at state s. We thus call the term $R(s, s') + \gamma * V^{\pi}(s') - V^{\pi}(s)$ as temporal difference (TD) term or TD error..

Once again, assume a discount factor γ=0.95 and a reward for non-terminal transitions is –0.04. Assume a learning rate α = 0.1.

1. Following steps below to update the utility using TD method.
    a. Begin by initializing all values to 0:

			+1
	X		-1

    b. Next, using α=0.1 and a first trial of:

$(1,1)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (1,2)_{\textbf{-.04}} \rightsquigarrow (1,3)_{\textbf{-.04}} \rightsquigarrow (2,3)_{\textbf{-.04}} \rightsquigarrow (3,3)_{\textbf{-.04}} \rightsquigarrow (4,3)_{\textbf{+1}}$

Carry out the series of state transitions and back-propagate the resulting cell values.

You need to update the table at one position each time an action is taken from this position. For example, for the first transition between (1,1) and (1,2), s is (1,1), s’ is (1,2), reward is -0.04, learning rate alpha is 0.1, and the discounting ratio is 0.95. We read the value of s and s’ from the newest updated table. Then we can update the utility at (1,1) using

$V^{\pi}(s) + \alpha * (R(s,s') + \gamma * V^{\pi}(s') - V^{\pi}(s))$
$= 0 + 0.1 * (-0.04 + 0.95 * 0 - 0)$
$= -0.004$

We then replace the value at (1,1) by -0.004.

Please follow this example to update the table for the sub-sequence of (**1,1) -> (1,2) -> (1,3)->(1,2) in the first trial.**

Please note that we do not wait until the episode/trial is over to update the table. In TD learning, we update the value at state s as soon as s transits to s’.

0 + 0.1 * (-0.04 + 0.95 * -0.004 - 0) = **-0.00438,**			+1
0 + 0.1 * (-0.04 + 0.95 * 0 - 0) = **-0.004,** **-0.004** + 0.1 * (-0.04 + 0.95 * -0.**00438 + 0.004**) = -0.0080161	X		-1
-0.004			

c. Again, using α=0.1, calculate the values at (1,1) only in a second trial:

$(1,1)_{\textbf{-.04}} \rightsquigarrow (2,1)_{\textbf{-.04}} \rightsquigarrow (3,1)_{\textbf{-.04}} \rightsquigarrow (3,2)_{\textbf{-.04}} \rightsquigarrow (4,2)_{\textbf{-1}}$.

			+1
	X		-1

-0.004 + 0.1 * (-0.04 + 0.95 * 0 + 0.004) = -0.0076			

20. Is TD method passive RL? Is TD model based or model free?

- Yes, TD uses reinforcement learning framework
- Yes, TD is Model Free

21. What is the advantage of using TD than direct utility estimation to estimate the U?

- one trial contains multiple samples if it is visited more than one times. so each sample is cumulative rewards. whereas in directly utility estimation, we take average if there are multiple sample. so in td, the estimation is much more stable.

22. What is the impact of alpha during the TD update? Why do we call alpha the learning rate? What does alpha mean when it is 0 or 1?

- A value closer to 1 would require a shorter time of evaluation however produce less stable results. Think: Tuning aggressively OR forget memory and accept ENTIRE change
- A value closer to 0 would require more time for evaluation with little to no “learning” in the decision making process. Takes previous value.

23. Do you think we should keep alpha fixed as more and more trials are executed? If not, how do you adjust the value of alpha as the number of trials increases?

As the number of trials increases, alpha should decrease

24. bootstrapping is a re-sampling technique in statistics. For example, in order to infer the mean, we could sample with replacement from the original dataset and compute the mean for each resample, then we could use the distributions of means to infer the variance of the population mean. It is called bootstrapping because it is like pulling yourself up using your own data. Do you think TD learning uses a similar bootstrapping idea?

- yes. because we are learning from what we have instead of bringing in new data. we compute a value for the same state by using previous value from same and other states.

25. Does TD give a better policy?

- No: HOWEVER Faster and most table (efficient) than other Model Based Reinforcement Learning methods

26. Why call this method “tempora difference”? Think about the recursive formula:

$V^{\pi}(s) \ = \ V^{\pi}(s) \ + \ \alpha \ * \ (R(s, s') \ + \ \gamma * V^{\pi}(s') \ - V^{\pi}(s) \ )$

- Error function evaluation: temporarily updating decision making process based on the difference between current and previous state values