Overview of Reinforcement Learning

Nikunj Rathod

M.Tech (ICT)- Machine Learning

Mentor : Prof. Abhishek Jindal

Dhirubhai Ambani Institute of Information and Communication Technology

Gandhinagar, 382007

Email: 202211014@daiict.ac.in

***Abstract* – This report shows how Reinforcement Learning is useful in various areas. Reinforcement Learning is a concept similar to Machine Learning which includes Agent, Environment and rewards. Reinforcement Learning can be applied in various fields such as security, games, robotics, medical, etc. This paper includes one of the game named "frozen lake" and explaining how Reinforcement Learning is applied to the game. How the agent takes actions based on the current state and gets rewards either negative or positive and learns accordingly for the future actions. Based on the findings, the report recommends further research in areas such as transfer learning, multi-agent RL, and deep RL to address these challenges.**

Index Terms – Agent, State, Environment, Action, Reward.

I. INTRODUCTION

Reinforcement Learning is somewhat similar to Machine Learning but different from the supervised learning where the model is trained from the labeled dataset. Reinforcement Learning includes Agent, Action and Environment. Here, agent is the user in the Environment and taking some actions and based on the actions, agent gets a numerical reward. Based on the rewards it gets, the agent learns if the action was worth taking or not. The reward is the kind of feedback to the user (here agent). The reward can be positive or negative. If the reward is positive, it means it is beneficial to the user and the action it has taken in good to take in future. But if the user gets negative reward, it means the action taken was not so good and user will avoid taking that action in the future. The ultimate goal of the system is to maximize the future reward.

Here in Figure 1, we can see the basic Diagram of Reinforcement Learning where Agent takes an Action (A_t) in the Environment from the current State (S_t) and according to the action gets a Reward (R_t) and goes to the next state(S_{t+1}).

Let's understand each term of Reinforcement Learning.

Environment : In which the agent operates and interacts with to learn and make decisions.
Agent : An agent is an entity that interacts with the environment and learns to make decisions based on feedback in the form of rewards or penalties.
State : Position of an Agent at a given time T.
Action : Agent's choice of activity in state S.
Reward : Price of taking right/wrong action.

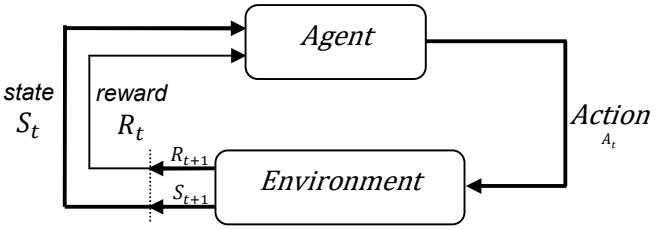

Figure1. Basic Diagram of Reinforcement Learning [1]

Let's break down this diagram into steps.

1. At time **t**, the environment is in state $\boldsymbol{S_t}$.
2. The agent observes the current state and selects action $\boldsymbol{A_t}$.
3. The environment transitions to state $\boldsymbol{S_{t+1}}$ and grants the agent reward $\boldsymbol{R_{t+1}}$.
4. This process then starts over for the next time step, **t+1**.

- Note, **t+1** is no longer in the future, but is now the present. When we cross the dotted line on the bottom left, the diagram shows **t+1** transforming into the current time step **t** so that $\boldsymbol{S_{t+1}}$ and $\boldsymbol{R_{t+1}}$ are now $\boldsymbol{S_t}$ and $\boldsymbol{R_t}$.

II. ELEMENTS OF REINFORCEMENT LEARNING

Beyond the above elements, there are other main sub elements of reinforcement learning system: A policy; A Reward function and Value function.

Policy defines the learning agent's way of behaving at a given time and state. Simply it is about how the agent will act in a given state at a given time. The action it takes is dependent on the policy it follows. Policy can be a simple function or table. In general, Policy may be stochastic.

Reward function is the function for calculating the reward that is received by the Agent. Basic goal of the Reward function is to maximize the total Reward it receives. Basically, the function assigns the numeric value (either positive or negative) to the state-value pair. This function defines what the good and bad events are for the agent. For example, If the agent gets a low or negative reward for taking an action following policy, then the policy may be changed to select some other action in the future in the same situation.

Value function is the term to specify what is good for the long run. It is total expected reward for the agent in the future, starting from the current state. We seek actions that bring out the highest value, not highest reward, because action obtains the greatest amount of reward for over the long run.

III. The Markov Property

The process of selecting an action from a given state, transitioning to a new state, and receiving a reward happens **sequentially** over and over again.

Reward Function

We saw that the agent's objective is to maximize the expected return of rewards it receives. If the sequence of rewards it receives after time step t is denoted by $r_{t+1}, r_{t+2}, r_{t+3}, \cdots, r_T$ then, the return is the sum of the rewards:

$$\boldsymbol{R_t = r_{t+1} + r_{t+2} + r_{t+3} + \cdots + r_T}$$

Where T is the final time step.

Another concept comes here is that of a discounting. The agent tries to select the actions so that the sum of the discounted rewards it receives over the future is maximized. It chooses A_t to maximize the expected discounted return:

$$\boldsymbol{G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots}$$

$$\boldsymbol{= \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}}$$

where γ is the discounting factor having value $0 < \gamma < 1$.

The discounted rate determines the present value of the future rewards. A reward received k time steps in the future are worth only γ_{k-1} times what it would be worth if it were received immediately.

$$\boldsymbol{G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots}$$

$$\boldsymbol{G_t = r_{t+1} + \gamma\left(r_{t+2} + \gamma r_{t+3} + \gamma^2 r_{t+4} + \cdots\right)}$$

$$\boldsymbol{G_t = r_{t+1} + \gamma G_{t+1}}$$

If $\gamma < 1$, the infinite sum has a finite value as long as the reward sequence $\{r_k\}$ is bounded. If $\gamma = 0$, the agent is being concerned only with maximizing the immediate reward, not the future rewards. As γ approaches 1, the objective takes future rewards into consideration more strongly: the agent become more farsighted.

$$\boldsymbol{G_t = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma}}$$

Transition Function

By applying action $a \in A$ in a state $s \in S$, the system makes a transition from s to a new state $s' \in S$, based on a probability distribution over the set of possible transitions.

The system being called is Markov if the result of an action does not depend on the previous actions and visited states (history), but only depends on the current (last) state, i.e.

$$\boldsymbol{P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots) = P(s_{t+1} \mid s_t, a_t)}$$

$$\boldsymbol{= T(s_t, a_t, s_{t+1})}$$

The idea of MDP is that the current state s gives enough information to make an optimal decision; if is not important which states and actions preceded s. Another way of saying this, is that if you select an action a, the probability distribution over next states is the same as the last time you tried this action in the same state.

Policies

Policy is defined as how probable it is for an Agent to take a specific action from any given state. It is a function that maps probabilities of selecting each possible actions from a given state and it is denoted by $\boldsymbol{\pi}$.

By saying that Agent is following the policy $\boldsymbol{\pi}$, We elaborate that the probability of Agent taking an action A from the given state S at time t.

$\boldsymbol{\pi(a|s)}$, where $A_t = a$ *and* $S_t = s$

For each state $s \in S$, $\boldsymbol{\pi}$ is a probability distribution over $a \in A(s)$.

State Value Function

The State Value Function is defined as how good the current state is for Agent following the policy $\boldsymbol{\pi}$. It gives us the value of the state under $\boldsymbol{\pi}$. The value of state ***s*** under policy $\boldsymbol{\pi}$ is the expected return from starting from state *s* at time *t* and following policy $\boldsymbol{\pi}$ thereafter. Mathematically we define $\vartheta_\pi(s)$ as

$$\vartheta_\pi(s) = E_\pi[G_t \mid S_t = s]$$

$$= E_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s\right]$$

State Value Function

The action-value function for policy $\boldsymbol{\pi}$, denoted as q_π tells us how good it is for the agent to take any given action from a given state while following policy $\boldsymbol{\pi}$. In other words, it

gives us the value of an action under $\boldsymbol{\pi}$. The value of action $\boldsymbol{a}$ in state $\boldsymbol{s}$ under policy is the expected return from starting from state $\boldsymbol{s}$ at time $\boldsymbol{t}$, taking action $\boldsymbol{a}$, and following policy $\boldsymbol{\pi}$ thereafter. Mathematically, we define $q_\pi(s,a)$ as

$$q_\pi(s,a) = E_\pi[G_t \,|S_t = s, A_t = a]$$

$$= E_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s, A_t = a\right]$$

The above mentioned function is called Q-function and the output of the function is called the Q-value. The letter "Q" represents the quality of taking an action in a given state.

IV. Implementation

Reinforcement leaning is commonly used in gaming area. I learned a game named frozen-lake in which reinforcement learning is applied. I studied that game and implemented the code it in the python language [5].

The description of the game is *"Winter is here. You and your friends were tossing around a frisbee at the park when you made a wild throw that left the frisbee out in the middle of the lake. The water is mostly frozen, but there are a few holes where the ice has melted. If you step into one of those holes, you'll fall into the freezing water. At this time, there's an international frisbee shortage, so it's absolutely imperative that you navigate across the lake and retrieve the disc. However, the ice is slippery, so you won't always move in the direction you intend. The surface is described using a grid like the following:"*

S FFF
FHFH
FFFH
HFFG

Here, we have a 4x4 grid. Each grid is a state in terms of reinforcement learning. We have total of 16 states in our game.

Here, S is out agent's starting state and G is the destination, where F represents the frozen surface and it is safe to put leg on the F surfaces while H represents Hole in the ice and it is dangerous to the agent. The agent will receive the reward of +1 if it successfully reached to the G state without getting into the hole. The episode ends when the agent either reaches to the Goal or fails into the Hole and next episode starts.

For implementing this game, I have used OpenAI's gymnasium library, simply called as gym. Gym library provides wide range of gaming environments used by reinforcement learning from basic games to advanced level games.

Here is the snapshot of the output of the game played.

```
  (Down)
SFFF
FHFH
FFFH
HFF
<ipykernel.iostream.OutStream object at 0x7f6fdc6c9a20>
****You reached the goal!****
```

Image1. Output of the frozen-lake game played. [5]

After few experiments, I observed that the maximum rewards are being given with the following values of the hyper-parameters.

Hyper parameter	Value
num_episodes	10000
max_steps_per_episode	99
learning_rate	0.01
discount_rate	0.95
exploration_rate	1
Max_exploration_rate	1
Min_exploration_rate	0.001
Exploration_decay_rate	0.001

Table 1. Hyper-parameter values for maximum reward. [5]

Acknowledgment

I would like to express my gratitude to all those who have contributed in this report on Applications of Reinforcement Learning.

Firstly, I would like to thank my mentor, Prof. Abhishek Jindal sir, for providing valuable guidance and continuously supporting the learning process. Their experience and knowledge were useful in the mathematical part of the learning process.

I am grateful for the contributions of all those who have made this project possible and hope that it will provide a useful resource for researchers, practitioners, learners and anyone enthusiastic in the field of reinforcement learning.

References

[1] Reinforcement Learning: An Introduction, Richard S. Sutton and Andrew G. Barto [A Bradford book; Cambridge, Massachusetts London, England]

[2] Deeplizard Reinforcement Learning Series[Online https://deeplizard.com/learn/video/nyjbcRQ-uQ8].

[3] Deepmind Reinforcement Learning material [https://www.deepmind.com/learning-resources/reinforcement-learning-lecture-series-2021].

[4] Scholar Paper [https://www.jair.org/index.php/jair/article/view/10166/24110]

[5] My github repo [https://github.com/NikunjRathod200/reinforcementLearning]