Assignment-15 : Reinforcement Learning

Harshvardhan Patidar
Department of Artificial Intelligence
Indian Institute of Technology Hyderabad
ai24btech11015@iith.ac.in

1 Reinforcement Learning

1.1 Introduction

The nature of learning brings to us the idea of learning by interacting with the environment. For learning, we need to have a sensorimotor connection or simply a connection between our agent and the environment which allows the agent to get information about the environment and to act upon the environment using some actuators. Using this connection, we try to gain information about cause and effect, that is the consequence of actions, and which action leads ultimately to our goal. The approach of reinforcement learning is also somewhat similar to this. Here, we focus on goal-directed learning via 'interaction' with the environment, rather than the typical approaches used in other fields of machine learning, where the set of actions that can be performed was predefined. In RL, the agent can explore and obtain information about various actions and its consequences.

1.2 Examples

These examples will help you understand the nature of reinforcement learning

- A master chess player's move depends upon both, possible replies by the opponent and the desirability of a particular move or position.
- An adaptive controller controls the petroleum refinery's operation based on real-time requirements.
- A new-born calf, which struggles even to stand, starts running after 20 minutes of birth.
- A mobile robot device deciding whether to explore new rooms or retreat back to recharge its battery, depending upon previous performance of battery.
- For a person preparing breakfast, he slowly improves and prepares better breakfast after he learns, according to his level of hunger, nutritional requirements, food preferences.

In all these examples, there is an agent, which interacts with the environment, takes action, which in turn affects the future state. Correct choice making requires to consider the effects of the previous actions, which means that it is supposed to do some foresight and planning before making its next move.

1.3 Elements of Reinforcement Learning

The main elements of a RL system are : a *policy*, a *reward signal*, a *value function* and sometimes a *model* of the function.

Policy is a mapping from the current state to the actions to be taken. It may be a simple function or a lookup table or might be involving extensive computations. It is alone sufficient to determine the behaviour of a RL model. It is generally stochastic, that is, specifies probability for each action.

Reward signal helps define the goal of a RL model. The environment sends a number, called a reward at every time step. The goal of a RL model is to maximize the sum of the rewards that it receives from the environment. The reward will be more for a step or action in the direction of achieving the goal as compared to some other move. Biologically, it may be considered analogous with pleasure and pain. They immediately define what the person is currently facing.

Value function is similar to reward. The only difference between them is while reward tells which action is better in an immediate sense, value function tells which state is better in the long run. Roughly speaking, the value of a state is the total amount of reward that an agent can expect to accumulate over the future steps, starting from that state. Even though we want to maximize the rewards for a function, we take decisions based on the value function.

Model of the environment is a part of some RL models called as *model-based* agents. This model is used for planning the next moves for a particular state and its expected reward and value. Given a state and action, using the model, we can predict what will happen and try to modify our action accordingly to get a better outcome.

1.4 Limitations and Scope

There are many evolutionary methods like the genetic algorithm, which do not estimate the value of some hypothesis function, instead they test multiple policies and pick the best performing one out of them. They do not actually interact with the environment. This renders them inefficient for many real world Reinforcement learning problems. RL focuses on interaction with the environment, which is more effective.

1.5 An Example : Tic-Tac-Toe

We will consider this example of Tic-Tac-Toe to illustrate RL and differences between RL and other approaches to develop a tic-tac-toe playing agent. We assume that we're playing against a less skilled opponent, and that both losses and draws are equally punishing for us.

One might think of classical ways to make such an agent. The "minimax" is a classical solution from game theory which can be implemented. But it would not be very useful, as it expects the opponent to play in a certain manner. One other method, of dynamic programming, is also not useful in this scenario as it requires a complete specification of the opponent, and probability with which it makes moves.

Evolutionary methods are applied, then they would directly search in the possible policies and use the one which has the highest probability of winning. For each policy, its probability of winning will be approximated by playing some games against the opponent. Such a model would successively generate and evaluate policies to obtain improvements.

But when RL, the main player kicks in, the decisions begin to be made on the basis of a value function. We first make a table, which has a number corresponding to each state, which represents our probability of winning from that state. If we play Xs, then all the states that have three Xs in a row have a probability of 1, all those which have three Os in a row have 0 probability of winning, whereas all other states are assigned a probability of 0.5. According to the table, the state which has higher probability of winning is considered better. Now to actually play, we will examine the state we will reach for our every possible move, and look for their current probability in the table. Then we will greedily select the move with the highest winning probability. We might also go with some other move which might not have the highest probability of winning, to *explore* the states we might never see otherwise.

As we proceed in the game, we update the table with more accurate estimates of probability of winning. More precisely, the current value of the state is updated to make it more closer to the latter state. Let S_t be state before the move, S_{t+1} after the move, and $V(S_t)$ be the value function, then we can update like below

$$V(S_t) \leftarrow V(S_t) + \alpha \left[V(S_{t+1}) - V(S_t) \right]$$

Here α is the step-size parameter, which determines the rate of learning. This update rule is used in the *Temporal-difference* learning method. This method converges to the true probabilities of winning.

This example thus explains the difference between RL and evolutionary learning methods. While the evolutionary ones neglect individual contributions and only considers the final outcome if we won or lost the match, RL on the other hand, values each individual state while playing, which makes it more efficient. The application of RL is not limited to the game of tic-tac-toe but is very much widespread. It is applicable for continuous cases as well, though it becomes more complex. Here, tic-tac-toe has a small finite set, but RL can be used even when the set is very large, or even in fact infinite. For example in the game of backgammon, which has approximately 10^{20} states.

Here we started just with the knowledge of the rules of the game, but in some other cases if we have prior knowledge, then that can very easily be incorporated into our model. In tic-tac-toe, the complete state is accessible, but RL can be applied in cases where the state is partially accessible. In a nutshell, what we saw about RL in this application is not the complete picture. RL is a lot more powerful than this.

1.6 Summary

It is clear from our understanding till now that RL is the first idea that comes into our mind when we think about making a machine learn in a goal-oriented way, and decision-making through interaction with the environment. It follows the Markov decision process framework, using states, actions, and rewards to represent the problem. The value and value functions are the key concepts which differentiates RL from the evolutionary learning methods.