Reinforcement Learning - Foundational concepts, applications and future trends

MARIAM MUHAMMED

INTRODUCTION

What is reinforcement learning?

Reinforcement learning is a type of machine learning that focuses on finding the best solution for an agent to map a particular environment to an action so as to increase the overall amount of reward out of the entire process. There are two main differentiating factors of reinforcement learning, which include **delayed reward** and **trial and error** approach to search. Reinforcement learning include an agent, an environment, action and reward. The agent in the process of performing the actions learns from the outcome of the interaction.

The agent continuously communicates with the environment, makes some actions and obtains feedback in the form of rewards or penalties in an attempt to discover the path that yields maximum reward in the long-run. Reinforcement learning is another of the three types of the learning algorithm, which is also known as the third generation of machine learning.

To gain a large number of rewards, the reinforcement learning agent prefers the action set which it has used in the past and that was successful in generating outcomes.

INTRODUCTION

Sub-elements of reinforcement learning

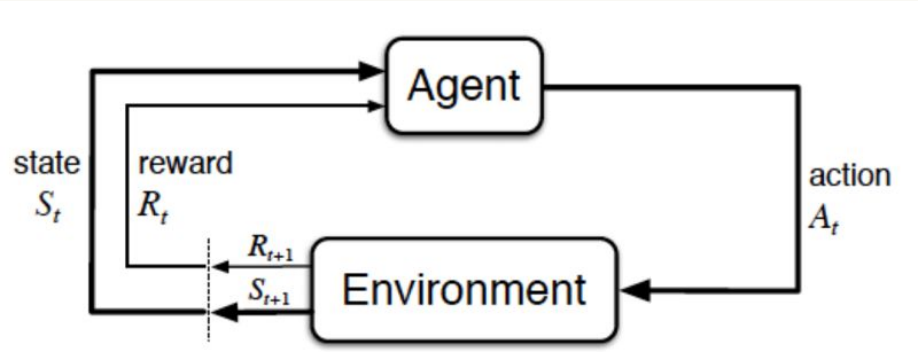
Beyond the agent and the environment, four main sub-elements of reinforcement learning are a policy, a reward, a value function and a model of the environment are:

- **Policy —** A policy is a state-to-action mapping. Policies are at the core of reinforcement learning.
- **Reward signals—**Reward signals, on the other hand, are the end product of reinforcement learning. The reward signals help agents differentiate between good and bad events. Typically, if an action selected by an agent is followed by a low reward, the policy is altered in the future so that a different line of action is taken instead
- **Value function —** A value function is different from a reward in the sense that a value function is focused on the long-term benefit while the reward is short-term
- **Model—**imitates the behaviour of the environment or can be seen as the blueprint of the environment

REINFORCEMENT LEARNING VS SUPERVISED LEARNING VS UNSUPERVISED LEARNING

REINFORCEMENT LEARNING	SUPERVISED LEARNING	UNSUPERVISED LEARNING
Reinforcement learning involves training an agent to make a sequence of decisions by interacting with an environment.	Learning from a training set of labelled examples provided by a knowledgeable external supervisor	Training a model on a dataset without labeled outputs. The model tries to learn the underlying structure or distribution of the data
The agent learns by receiving rewards or penalties based on its actions.	Extrapolate responses so that it acts correctly in situations not present in the training set to	find hidden patterns, groupings, or representations in the data
Gaming, Robotics	Classification, regression(e.g price prediction)	Clustering

MARKOV DECISION PROCESS (MDP)



The Markov property which postulates that the current state of the environment is dependent only on the current state and the action taken at that particular time and not the sequence of events that occur leading to that time is another postulate that underlines Reinforcement Learning (RL).

This feature is critical for RL because it simplifies the modeling and forecasting of the actions of the environment, which can be helpful in both learning and decision-making processes.

MARKOV DECISION PROCESS (MDP)

The MDP and agent together result in actions shown in equation 3.1 below

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \cdots . \qquad eqn\ 3.1$$

Equation 3.2 defines the likelihood of transitioning to state s' with reward r from state s after taking action a. This must satisfy the property of total probability, i.e., the sum over all possfollowing following states and rewards is 1(eqn 3.3), ensuring a well-defined probability distribution.

$$p(s', r \mid s, a) = \Pr\{ S_t = s', R_t = r \mid S_{t-1} = s , A_{t-1} = a\}. \qquad eqn\ 3.2$$

$$\sum_{s' \in S} \sum_{r \in R} p(s', r \mid s, a) = 1, for\ all\ s \in S, a \in A(s). \qquad eqn\ 3.3$$

In reinforcement learning the purpose or goal of any agent is formalised in terms of a special signal called reward, passing from the environment to the agent.

DELIVERY ROBOT MDP modelling

How does the delivery robot decide on the action to take?

States

- S = {charged,low}

Actions

- $A_{charged}$={deliver,wait}
- A_{low}={deliver, wait, recharge}

Probabilities

ɑ - probability that search begins with high and ends with high
1- ɑ - probability that search begins with high and ends with low
β - probability that search begins with low and ends with low
1 - β - probability that search begins with low and ends with high
-1 - probability that the agent needs to be recovered

s	a	s'	p(s'|s,a)	r(s,a,s')
charged	deliver	charged	ɑ	r_{search}
charged	deliver	low	1 - ɑ	r_{search}
low	deliver	charged	1 - β	-1
low	deliver	low	β	r_{search}
charged	wait	charged	1	r_{wait}
charged	wait	low	0	-
low	wait	charged	0	-
low	wait	low	1	r_{wait}
low	recharge	charged	1	-
low	recharge	low	0	-

MONTE CARLO METHODS

Monte Carlo method is a learning method for estimating the value functions and discovering optimal policies. Monte Carlo methods only require experience, a collection of sample states, actions and rewards from actal or simulated interaction with an environment. Reinforcement learning problems are solved based on averaging sample returns. Monte Carlo estimation encompasses the general approach of policy iteration, which alternates between evaluating and improving a policy.

Equation 3.7 represents a sequence where:

- π_0 is an initial policy.
- $q\pi_0$ is the value function derived from evaluating policy π_0, likely using a Monte Carlo method where returns are averaged over multiple episodes.
- π_1 is an improved policy, typically formulated by making greedy choices based on $q\pi_0$.
- This cycle continues until convergence to an optimal policy π_* and its corresponding value function q_*.

$$\pi_0 \xrightarrow{E} q\pi_0 \xrightarrow{I} \pi_1 \xrightarrow{E} q\pi_1 \xrightarrow{I} \pi_2 \xrightarrow{E} \cdots \xrightarrow{I} \pi_* \xrightarrow{E} q_* \ , \quad eqn\ 3.7$$

At any state s, the chosen action a is the one thatmaximisess the state-action value (s,a) $q(s,a)$ as shown in eqn 3.8. This forms the basis of a greedy policy, where decisions are made purely on current estimations of value functions.

$$\pi(s) = \arg\max_a q(s,a). \qquad eqn\ 3.8$$

TEMPORAL DIFFERENCE LEARNING

Temporal difference(TD) learning is central and novel to reinforcement leanring. TD can learn from raw experience without a model of the environment. The core equations behind the TD methods are explained below:

Equation 3.1 explains the basic TD update where (St) is updated towards the actual return Gt observed after state St. Here, α is the learning rate. This method is typically used in Monte Carlo learning where Gt is the return from state St until the end of the episode.

$$V(S_t) \leftarrow V(S_t) + \alpha[G_t - V(S_t)], \qquad eqn\ 3.11$$

This updates the value function based on the reward received at the next timestep Rt+1, plus the discounted value of following next state $\gamma V(St$+1), minus the current value estimate.

$$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)] \qquad eqn\ 3.12$$

POLICY GRADIENT METHOD

Policy gradient methods are widely used in the reinforcement learning algorithms owing to their inherent ability to directly optimise the sum of rewards and their applicability with various high-order nonlinear approximating function such as neural network John et al. (2015).

These methods have been extrapolated in reducing high variance gradient estimation which is typical of policy gradients as derivable from the work done by Sehnke et al. in PGPE. New families of policy values or policy derivative methods which include Proximal Policy Optimization Algorithms have also been developed to overcome the exploiter-explore duality in reinforcement learning (Schulman, 2017).

APPLICATIONS OF REINFORCEMENT LEARNING

- **Robotics and Autonomous Systems**

  Significant use of Reinforcement Learning (RL) has been observed in autonomous systems and robotics. In the context of multiagent systems, RL allows each agent to learn how best they can navigate their environments on their own, thus enhancing the possibility of coordination between such agents

- **Game Playing and Strategy Development**

  As a relatively recent subfield of machine learning, RL has often been used in games, especially in playing and strategy formation, thus making it evident that it is capable of mastering games as well as coming up with superior strategies for such functions.

- **Recommendation Systems**

  RL has progressively been used in recommendation systems in order to handle the difficulties associated with personalization, negative feedback, and proactive content recommending.

- **Healthcare and Personalised Treatment**

  Reinforcement learning (RL), one of the promising fields of AI applications, has recently been implemented in the health care and-individualized treatments, where it drastically changes the existing approach to the medical treatment and patient care. The references also help in gaining understanding of uses of RL in provinces of health care which includes individualized treatment, promoting physical activity and variable treatment recommendation.

CHALLENGES

- **Safety and Robustness**

  Safety constraints play an important role when incorporated in RL particularly for those that deal with physical interaction. Existing RL algorithms have failed to incorporate sound measures for safe actions especially in light of uncertain and ever-changing situations.

- **Exploration vs. Exploitation**

  Balancing exploration and exploitation is inherently challenging in RL. Adequate exploration is necessary to discover optimal policies, but excessive exploration can lead to suboptimal learning and performance. Designing strategies that effectively balance these aspects is critical for the success of RL algorithms.

- **Sample Efficiency**

  One of the most significant challenges in RL is sample efficiency. RL algorithms typically require many interactions with the environment to achieve competent performance. This is impractical in scenarios where interactions are costly or time-consuming.

- **Stability and Convergence**

  The stability of learning updates and the convergence of algorithms are not guaranteed in non-linear function approximators like neural networks. This unpredictability can lead to divergent behaviours and unstable learning processes.

FUTURE DIRECTION

- **Model-based Reinforcement Learning**

  Enhancing the sophistication of model-based techniques could further enhance the sample efficiency and stability. While learning models of the environment, the RL algorithms are able to make optimal plans for action and could need less attempts in the real world.

- **Safe Reinforcement Learning**

  Safety and robustness metrics should be designed, and incorporated into the RL framework to enhance the system. Safe RL involves developing methods that can meet stringent safety requirements as it pertains to learning and deployment.

- **Multi-agent Systems**

  Studying the multi-agent systems might help to identify the approaches towards decision-making in the distributed environment and cooperation schemes. These systems are beneficial in cases with a number of decision-makers and can contribute to the development of sound and scalable RL approaches.

- **Integration with Other Learning Paradigms**

  Integration of RL with other learning approaches like unsupervised and supervised learning was found to have drawbacks in exploration and stability. One could then use the elements of each framework in a complementary manner to improve overarching learning effectiveness.