Reinforcement Learning

Leonidas Avdelas

1 Introduction

Reinforcement Learning (RL) is defined as having an agent learn what to do — how to map situations to actions — so as to maximize a numerical reward signal. The agent is not told which actions to take, but instead must discover which actions yield the most reward by trying them out [Sutton and Barto, 2018]. Moreover, in many cases the actions of the agent do not only affect the immediate reward, but also the next situation and possibly all subsequent rewards. To achieve that, RL aims to solve problems through *trial-and-error* in an environment with *delayed reward*.

The field of RL has its roots on two areas. The first one is behaviorist psychology, where the paradigm of trial-and-error learning originates from, and the second one is optimal control, which has lent the mathematical formalisms (most notably dynamic programming) that underpin the field.

RL is some times confused with supervised learning. The main difference between supervised learning and RL is that on supervised learning, the model is trained based on samples and labels and each guess is considered a unique event. On the other hand in RL, as mentioned, there may be a lot of steps before the agent learns if the decision taken is correct, and might never find out what the exact true/best value was, but only the impact the agent's actions had on the environment.

2 General

More formally, in an RL environment, an autonomous agent, controlled by a machine learning algorithm, observes a state s_t from its environment at time step t. The states originate from a sate space $\mathcal{S}$. The agent interacts with the environment by taking an action a_t in state s_t, picked from an action space $\mathcal{A}$. When the agent takes an action, the environment and the agent transition to a new state, s_{t+1}, based on the current state and the chosen action [Arulkumaran et al., 2017]. The agent also receives a scalar reward R_t deriving from the state-action pair. This reward acts as a form of feedback for the agent's action. Additionally, the agent

keeps a state to action mapping, based on which it makes decisions. This sate-action mapping is called the agent's **policy** and is denoted as $\pi(a_t|s_t)$. For each state action pair, there is a related reward R that the agent receives for taking the particular action a_t on state s_t. The best sequence of actions is determined by these rewards provided by the environment. The goal of the agent is to learn a policy π that maximizes the expected return (cumulative, discounted reward). Given a state, a policy returns an action to perform; an optimal policy is any policy that maximizes the expected return in the environment.

We want RL problems to satisfy the Markov Property. That means that for each state, the future depends only on the current state and not the previous ones. Then we can model the problem as a Markov Decision Processes (MDPs), which consist of

- a set of states, $\mathcal{S}$, plus a distribution of starting states $p(s_0)$
- a set of actions $\mathcal{A}$
- the state-transition probabilities $\Pr\{S_t = s'|S_{t-1} = s, A_{t-1} = a\}$ that map a state-action pair at time $t-1$ onto a distribution of states at time t.
- an immediate/instantaneous reward function $R(s_{t-1}, a_{t-1}, s_t)$.
- When calculating the cumulative reward, a discount factor is included $\gamma \in [0, 1]$, which is used to place less emphasis on immediate rewards.

Based on the above, we define the return as the discounted, accumulated reward together with the discount factor:

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

A key component of every RL method is the **value function**, a prediction of the expected, accumulative, discounted, future reward. It measures how good each state, or state-action pair is. The state value,

$$v_\pi(s) = \mathbb{E}[R_t|s_t = s] \tag{2}$$

is the expected return for following policy π from state s. The action value,

$$q_\pi(s, a) = \mathbb{E}[R_t|s_t = s, a_t = a] \tag{3}$$

is the expected return for selecting action a in state s and then following policy π.

Moreover, to better describe RL methods we divide them in **model-based** and **model-free** problems. **Model-based** methods are used when we have a complete knowledge of the dynamics of the surrounding environment and rely on *planning*

as their primary component. The methods include dynamic programming and similar methods. We use policy evaluation to calculate the return of a specific policy and On the other hand **model-free** methods require no prior knowledge of the environment and rely on *learning*. By a *model* of the environment we mean anything that an agent can use to predict how the environment will respond to its actions.

References

[Arulkumaran et al., 2017] Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38.

[Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA.