Reinforcement Learning: Value Function and Policy

Sebastian Dittert
Apr 14, 2020 · 6 min read ★

In the last **article** I described the fundamental concept of Reinforcement Learning, the **Markov Decision Process (MDP)** and its specifications. With the help of the MDP, Deep Reinforcement Learning problems can be described and defined mathematically.

Since, as described in the MDP article, an agent interacts with an environment, a natural question that might come up is: **How does the agent decides what to do,**

what is his decision-making process? Further, the agent might want to know how good his actions have been and **evaluate his current situation in the environment, in the sense of wanting to solve the Problem?**

This is exactly what the following article will deal with. The concrete interaction between the agent and the environment. **How does the agent evaluate his temporary situation in the environment and how does he decide what action to take?**

For this purpose there are two concepts in Reinforcement Learning, each answering one of the questions. The v**alue function** covers the part of evaluating the current situation of the agent in the environment and the **policy,** which describes the decision-making process of the agent. Both shall be explained below…

Policy

A policy (π) describes the decision-making process of the agent. In the simplest case, the policy for each state refers to an action that the agent should perform in that state. This type of strategy is called **deterministic policy**. Each state is assigned an action, for example for state ***s1:*** π***(s1) = a1***. A **deterministic policy can be displayed in a table**, where an action can be selected in different states:

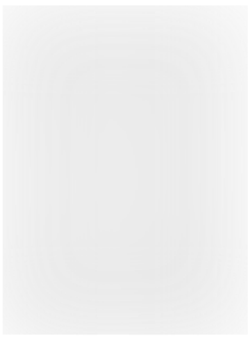


table of a deterministic policy

In general, a policy assigns probabilities to every action in every state, for example π***(s1|a1) = 0.3***. **The policy thus represents a probability distribution for every state over all possible actions**. Such a policy is called a **stochastic policy**. In a stochastic policy, several actions can be selected, whereby the actions each have a probability of non-zero and the sum of all actions is 1. Thus, it can be said that the behavior of an agent can be described by a policy, which assigns states to a probability distribution over actions. The policy is only dependent on the current state and not on the time or the previous states.

probability distribution for a stochastic policy

Value Function

The Value Function represents the value for the agent to be in a certain state. More specifically, the state value function describes the expected return ***G_t*** from a given state. In general, a state value function is defined concerning a specific policy, since the expected return depends on the policy:

The index π indicates the dependency on the policy. Furthermore an **action-value function** can be defined. The action-value of a state is the expected return if the agent chooses action ***a*** according to a policy π.

Value functions are critical to Reinforcement Learning. They allow an agent to query the quality of his current situation rather than waiting for the long-term result. This has a dual benefit. First, the return is not immediately available, and second, the return can be random due to the stochasticity of the policy as well as the dynamics of the environment. The value function summarizes all future possibilities by averaging the returns. Thus, **the value function allows an assessment of the quality of different policies**.

A fundamental property of value functions used throughout RL is that they satisfy **recursive relationships**. For each policy and state ***s***, the following consistency condition applies between the value of ***s*** and the value of its possible subsequent states:

This equation is also called the ***Bellman equation.*** For the Value Function the Bellman equation defines a relation of the value of State ***s*** and its following State ***s'***. The Bellman equation is also used for the Action-Value function. Accordingly, the Action-Value can be calculated from the following state:

In the Bellman equations the structure of the MDP formulation is used to reduce this infinite sum to a system of linear equations. By directly solving the equation, the exact state values can then be determined.

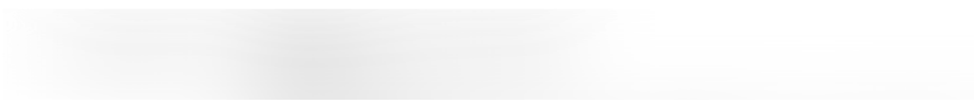
Optimal Policy

To solve a task or a problem in RL means to find a policy that will have a great reward in the long run. For finite MDPs, an optimal policy can be precisely defined in the following way. Value Functions define a partial order over different policies. For example, **a policy π is better or at least as good as a policy π′ if the expected return across all states is greater than or equal to that of π′.** In other words, π ≥ π′ is better for and only if ***v_pi* ≥ *v_*π′** is better for all states. This is an optimal policy **π∗.**

Although there may be several optimal policies, they all share the same state value function, which is called optimal state value function and is defined as follows:

Optimal policies also share the same optimal action-value function:

Due to the fact that **v∗** is a value function for a policy, it must meet the condition of uniformity of the Bellman equation. Since it is the optimal value function, the consistency condition of **v∗** can be written in a special form without reference to a specific policy. This Bellman equation for **v∗** is also called **Optimal Bellman Equation** and can also be written down for the optimal action-value function.



Once ***v∗*** exists it is very easy to derive an optimal policy. There will be one or more actions for each state ***s***, where a maximum in the optimal Bellman equation is reached. Any policy that assigns a probability greater than zero to only these actions is an optimal policy.

Using ***v∗*** the optimal expected long-term return is converted into a quantity that is immediately available for each state. A one-step predictive search thus yields the optimal long-term actions.

With ***q∗***, on the other hand, the agent does not have to perform a one-step predictive search. For each state s only one action has to be found, which maximizes ***q∗ (s, a)***.

Effectively, the action-value function combines all results of the single-stage predictive search. For each state-action pair, the optimal expected long-term return is displayed, allowing the selection o**f optimal actions without the knowledge of future states and their value, and thus without knowing anything about the dynamics of the environment.**

Important for Reinforcement is that both, policy, as well as value function/action-value function, can be learned and lead to a close optimal behavior.

For Deep Reinforcement Learning policy and value function can be represented as a neural network. Whereas both different strategies use to optimize their network parameters. This splits the field of model-free reinforcement learning in two sections: **Policy-Based Algorithms** and **Value-Based Algorithms.**

However, these are topics for a subsequent article and will not be explained here. Finally, I hope this article has helped you to understand the policies and value functions a little better.

Reinforcement Learning Machine Learning Deep Learning Artificial Intelligence

Decision Making