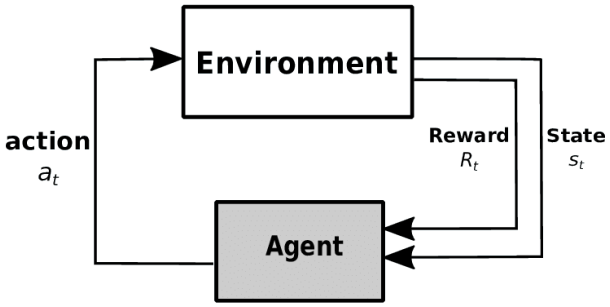
Introduction to Reinforcement Learning in PyTorch[2]

---

1 Basics of Reinforcement Learning

Figure 1: [1]



RL algorithms are often modeled as Markov Decision Processes. Hence, at time step t, the **agent** (RL algorithm) is situated in **state** s_t. The agent interacts with an environment by taking an **action** a_t. This action results in a new state s_{t+1} and the transition (s_t, a_t) brings with it a **reward** r_t. Often times, there is a probability distribution over the transition (s_t, a_t) to a new state s_{t+1}. Hence, each transition has a certain probability of ending up in each state. Additionally, there often exist **episode-ending states**, which corresponds to reaching a final goal. Your goal is to learn a **policy** π that maps states to actions. Although, in an MDP, we assume that we can always tell which state s_t our agents is in, this isn't always the case. In these cases, we have observations o_t. The goal of an agent is to maximize the total reward R. Therefore, it is important to ensure that the reward actually captures the true goal we want the agent to achieve/learn.

Often times, you want the agent to achieve its goal as soon as possible. In this case, one can apply the concept of **discounted rewards**: decreasing the reward that the agent receives over time to stimulate the agent to solve the problem as fast as possible. This is done by a time-dependent multiplicative term γ^t. With discounting then, the agent's goal is to maximize

$$\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t] \tag{1}$$

2 Notes

We always know the state of the agent $\rightarrow$ no observations necessary.

The only episode-ending state is when the assets of the agent becomes non-positive.

We do not want to use a discount factor, since it does not matter for a trading bot whether it makes a lot of money at time t or at time $t + 1$.

3 Notation

a_t	Action taken at time t.
r_t	Reward received by agent at time t.
s_t	State of the agent at time t.
(s_t, a_t)	Transition from state s_t by taking action a_t, resulting in reward r_t.
R	Total reward

4 Definitions

Episode: The trajectory of going from start to finish of a task.

References

[1] Roohollah Amiri et al. “A Machine Learning Approach for Power Allocation in HetNets Considering QoS”. In: (Mar. 2018).

[2] Harsh Panchal. *Introduction to Reinforcement Learning (RL) in PyTorch*. URL: https://medium.com/analytics-vidhya/introduction-to-reinforcement-learning-rl-in-pytorch-c0862989cc0e. (accessed: 26.08.2024).