REINFORCEMENT LEARNING

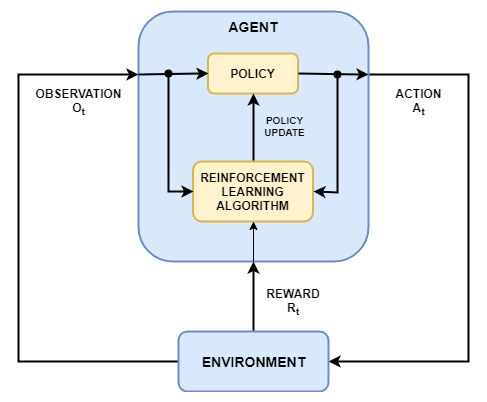


Figure : Reinforcement Learning Model

# POLICY

A **policy** is a decision-making strategy that guides an agent's actions in an environment. It defines the agent's behavior by specifying the probability of selecting a particular action given the current state. A policy can be either deterministic or stochastic.

where:

* is the policy,
* is the current state,
* is the action chosen in state .

Equation : Deterministic policy function

where:

* is the probability of choosing action a in state ,
* is the conditional probability of taking action given state .

Equation : Stochastic policy function

A **deterministic policy** maps each state to a single action. In other words, for every state, there is a specific action that the agent will always choose when following the deterministic policy. Mathematically, it can be represented as a function as Equation 1.

A **stochastic policy** maps each state to a probability distribution over the set of possible actions. This means that the agent chooses an action based on probabilities rather than a fixed mapping. Stochastic policies can help the agent explore the environment and avoid getting stuck in suboptimal solutions. Mathematically, a stochastic policy can be represented as Equation 2.

The goal in reinforcement learning is to find an optimal policy, which maximizes the expected cumulative reward for the agent over time. This optimal policy can be derived from the optimal value function or the optimal Q-function, depending on the specific RL algorithm used.

# Q-Learning

Q-learning is a model-free reinforcement learning algorithm that aims to learn the optimal policy for an agent interacting with an environment. It does this by learning a Q-function (also called a state-action value function) that estimates the expected cumulative reward for taking a particular action in a given state and then following a specific policy thereafter. The Q-function is denoted as , where represents the state and represents the action.

The learning process in Q-learning involves updating the Q-function iteratively using the Bellman equation, which expresses the relationship between the current state-action value and the expected value of the next state-action pair, taking into account the immediate reward received for performing the current action.

where:

* is the current state.
* is the action taken in the current state.
* is the immediate reward received after taking action in state . It's a crucial part of the update because it provides direct feedback about the action's value.
* is the next state (resulting from taking action in state ).
* is the action taken in the next state .
* (alpha) is the learning rate, which controls the step size of the update. A higher learning rate makes the agent more sensitive to recent experiences, while a lower learning rate makes the agent more conservative, relying more on its accumulated knowledge.
* (gamma) is the discount factor, which determines the importance of future rewards relative to immediate rewards.
* represents the maximum Q-value for the next state over all possible actions .
* represents the estimated cumulative reward from the next state () onward, assuming the agent follows the optimal policy. By taking the maximum Q-value over all possible actions () in the next state (), we assume that the agent will choose the action that leads to the highest expected future rewards.
* calculates the difference between the updated Q-value (taking into account the immediate reward and the maximum future reward) and the current Q-value. This difference, also known as the temporal difference (TD) error, measures how much our current estimate is off from the new information we just received.

The update rule combines the current Q-value with the weighted TD error to produce a new Q-value for the state-action pair. By incorporating both immediate rewards and future rewards, the agent updates its estimates to better reflect the true value of each action in a given state.

**Note:** This equation can be read as “Update the Q-value for state and action by assigning it the current Q-value plus the weighted temporal difference error.”

Equation : Update rule for Q-learning

In Q-learning, the agent iteratively updates its Q-function based on its experiences in the environment. The agent chooses actions using an exploration-exploitation strategy, such as ε-greedy or SoftMax, balancing the need to explore new actions and exploit the current knowledge of the Q-function.

As the agent interacts with the environment and updates its Q-function, the Q-values converge to the optimal Q-function, which represents the expected cumulative reward for each state-action pair when following the optimal policy. The optimal policy can then be derived from the optimal Q-function by choosing the action with the highest Q-value in each state.