TD 9 – Learning III Reinforcement Learning

1 Markov Decision Process

① *Return*

The return is the total future rewards :

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... = \sum_{k=0}^{+\infty} \gamma^k r_{t+k+1}$$

The recursive form is obtained by factorizing by γ :

$$G_t = r_{t+1} + \gamma(r_{t+2} + \gamma r_{t+3} + ...) = r_{t+1} + \gamma G_{t+1}$$

② *Value of an action in a state*

The subjective value of an action in a state is the expected return if the agent starts in state s_t, chooses the action a_t, and then follow its policy :

$$Q_\pi(a_t, s_t) = \mathbb{E}(G_t|a_t, s_t) = \mathbb{E}(r_{t+1} + \gamma G_{t+1} \mid a_t, s_t) = \mathbb{E}(r_{t+1} \mid a_t, s_t) + \gamma \mathbb{E}(G_{t+1} \mid a_t, s_t)$$

- Once an action a_t and a state s_t are fixed, the random variable r_{t+1} becomes determined by the reward function : $\mathbb{E}(r_{t+1} \mid a_t, s_t) = R(a_t, s_t)$
- The random variable G_{t+1} depends on the state s_{t+1}, which itself depends on the stochastic evolution of the environment to the next state s_{t+1} after the action a_t has been performed in the current state s_t :

$$\mathbb{E}(G_{t+1}|a_t, s_t) = \mathbb{E}\left(\bigcup_{s_{t+1}} (G_{t+1} \bigcap s_{t+1})|a_t, s_t\right) = \sum_{s_t} p_{tr}(s_{t+1}|a_t, s_t) \underbrace{\mathbb{E}(G_{t+1}|s_{t+1})}_{V_\pi(s_{t+1})}$$

Gathering both terms leads to the recursive expression :

$$Q_\pi(a_t, s_t) = R(s_t, a_t) + \gamma \sum_{s_{t+1}} p_{tr}(s_{t+1}|a_t, s_t) V_\pi(s_{t+1})$$

③ *Value of a state*

The value of a state is the expected return if the agent starts in state s_t and follow its policy :

$$V(s_t) = \mathbb{E}(G_t|s_t) = \mathbb{E}(r_[t+1] + \gamma G_{t+1}|s_t) = \mathbb{E}(r_[t+1]|s_t) + \gamma \mathbb{E}(G_{t+1}|s_t)$$

The expectancy is taken over all possible evolutions of the environment and all actions taken by the agent (as now both are stochastic).

- The random variable r_{t+1} depends on the stochastic choice of the action a_t in the current state s_t :

$$\mathbb{E}(r_{t+1}|s_t) = \sum_{a_t} \pi(a_t|s_t) R(s_t, a_t)$$

- The random variable G_{t+1} depends on the state s_{t+1}, which itself depends on the stochastic choice of the action a_t in the current state s_t and the stochastic evolution of the environment to the next state s_{t+1} :

$$\mathbb{E}(G_{t+1}|s_t) = \mathbb{E}\left(\bigcup_{s_{t+1}} (G_{t+1} \bigcap s_{t+1})|s_t\right) = \sum_{a_t} \pi(a_t|s_t) \sum_{s_t} p_{tr}(s_{t+1}|a_t, s_t) \underbrace{\mathbb{E}(G_{t+1}|s_{t+1})}_{V_\pi(s_{t+1})}$$

Gathering both terms leads to the recursive expression :

$$V_\pi(s_t) = \sum_{a_t} \pi(a_t|s_t) \left[R(s_t, a_t) + \gamma \sum_{s_{t+1}} p_{tr}(s_{t+1}|a_t, s_t) V_\pi(s_{t+1}) \right]$$

2 Algorithms for Decision-Making

2.1 Dynamic Programming

④ *Characterization of the optimal policy in terms of the distribution* $\pi^*(a|s)$

• In any state, the action chosen is the one which gives the maximum expected reward. Thus, the optimal policy becomes deterministic. Formally, for any state s, $\pi^*(a|s) = 1$ for the action $a^* = \arg\max_a Q^*(a, s)$.

• Thus, the optimal policy can be deduced straight-forward from the state-action value function Q^*.

⑤ *Characterization of the optimal policy with a relation between* $V^*(s)$ *and* $Q^*(a, s)$

• The expected reward of a state is the maximum reward since the actions chosen are optimal at each step (by definition of the optimal policy). Formally, $V^*(s) = \max_a Q^*(a, s)$.

• Thus, the optimal state-action value function Q^* can be viewed as the fixed point of a function f by rewriting the update formula :

$$Q^*(a, s) = R(a, s) + \gamma \sum_{s'} p_{tr}(s'|a, s) V^*(s') = R(a, s) + \gamma \sum_{s'} p_{tr}(s'|a, s) \max_{a'} Q^*(a', s')$$

⑥ *Numerical method for finding the optimal policy*

It is possible to use the iterative fixed point method.

2.2 Reinforcement Learning

⑦ *Two types of reinforcement learning models*

- Model-free agents (MF-RL) directly estimate their subjective values with experience at time t.
- Model-based agents (MB-RL) build models of the expected transitions $\widehat{P}$ and expected rewards $\widehat{R}$.

⑧ *Temporal-Difference Error*

• According to question ③ :

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

This expression rewrites $\delta_t = r_t - (V(s_t) - \gamma V(s_{t+1}))$, which is the difference between the expected reward $V^{(t)}(x_t) - \gamma V^{(t)}(x_{t+1})$ and the actual reward r_t.

⑨ *Online update of the value function*

The update can be made proportional to the error with a learning rate α :

$$V(s_t) \leftarrow V(s_t) + \alpha \left[r_t + \gamma V(s_{t+1}) - V(s_t) \right]$$

⑩ *Error and update in the Q-learning algorithm*

Error : $\delta_t = r_t - (Q^{(t)}(x_t, u_t) - \gamma \max_{u_{t+1}} Q^{(t)}(x_{t+1}, u_{t+1}))$.

Update : $Q(a_t, s_t) \leftarrow Q(a_t, s_t) + \alpha \delta_t$.

The difference between strategies is that in Q-Learning, an entry is updated selectively according to the action a_t taken in state s_t, whereas in TD-Learning an entry is updated whatever the action taken in state s_t.

⑪ *Soft-max policy*

The soft-max policy is governed by the following formula :

$$\pi(a_t|s_t) = \frac{e^{\beta\, Q(a_t, s_t)}}{\sum_{a'} e^{\beta\, Q(a', s_t)}}$$

The agent faces an 'exploration-exploitation' dilemma : it should choose between exploiting the current information it has about the environment and exploring new solutions. Such a formula is also found in statistical physics. β plays the role of an inverse temperature, which switches the balance between exploration and exploitation.

- As the parameter β increases (temperature decreases), the denominator sum gets dominated by its highest term such that the policy becomes dominated by a single value $Q(a_t|s_t)$, which is the one maximizing expected future rewards. This favors exploitation, with the issue that it relies only on the knowledge about the environment that the agent has gathered up to the current state, which might be incomplete due to stochastic sampling.
- As the parameter β decreases (temperature increases), all terms in the denominator sum become equivalent and π becomes more and more uniformly distributed. This favors exploration, with the issue that the agent does not commit to choose the best solution, and rewards are then random.

3 Applications to Maze Learning

3.1 Analytical study – Model-free agent performing Temporal-Difference Learning

⑫ *Model of the situation*

The states of the environments correspond to a set of positions in the corridor $\{-n, -n+1, ..., n-1, n\}$, where $-n$ is the left end of the maze, n is the right end of the maze, and 0 is the middle of the maze.

The actions that the agent can perform are displacements in both direction 'right' and 'left'.

The reward is 1 at the exit, 0 everywhere else.

⑬ *Value function $V(k)$ (after convergence)*

For any position $k \in [\![-(n-1), (n-1)]\!]$, two actions 'right' and 'left' are possible, each with probability $1/2$ and no reward. Therefore, $V(k) = \frac{1}{2}V(k+1) + \frac{1}{2}V(k-1)$.

At the extreme positions, a reward may or may not be received depending on the side, and exploration stops : $V(n) = 1$ and $V(-n) = 0$.

⑭ *Ansatz for solving the equations*

The previous equation corresponds to $\Delta V(k) = 0$.

In the continuous limit, this differential equation would entail that V is an affine function of k :

$\partial_k^2 V = 0 \implies V(k) = Ak + B$.

With an ansatz of this form and the boundary conditions :

$$\begin{aligned} V(-n) = 0 \quad &\Rightarrow \quad B = An \\ V(n) = 1 \quad \Rightarrow \quad An + B = 2An = 1 \quad &\Rightarrow \quad A = \frac{1}{2n} \end{aligned}$$

Therefore, $V(k) = \frac{1}{2}\left(1 + \frac{k}{n}\right)$.

⑮ *Estimated value V during the first learning trial* On the first trial, all positions are equivalent for the subjective values of the agent. The TD error is $R(k) + \widehat{V}(k \pm 1) - \widehat{V}(k) = 0$, except at the boundaries, such that no updates to $\widehat{V}$ are made until the agent reaches one of the ends of the maze. If the left end is reached, then the trial stops without any update. If the right end is reached, then an update is made at the position $n-1$ using the value of the reward. Progressively, with many essays, V will be updated from position to position, back-propagating from the right end of the maze.

3.2 Simulations – Model-based agent

⑯ *Python objects/structures*

- Reward function $R(a, s)$ and estimated reward function $\widehat{R}(a, s)$: 2D matrix of dimensions $|\mathcal{A}| \times |\mathcal{S}|$.
- Transition probabilities $p_{tr}(s_{t+1}|a_t, s_t)$ and estimated transition probabilities $\widehat{P}(s_{t+1}|a_t, s_t)$: 3D matrix of dimensions $|\mathcal{S}| \times |\mathcal{A}| \times |\mathcal{S}|$.
- State-action value function $Q(a, s)$: 2D matrix of dimensions $|\mathcal{A}| \times |\mathcal{S}|$.

- Number of events of each (state, action) couple $N(a, s)$: 2D matrix of dimensions $|\mathcal{A}| \times |\mathcal{S}|$.

⑰ – ㉔ **num**  Jupyter Notebook