M2 Cogmaster 2024–2025 | November 21st
TA: Hélène Todd (helene.todd@ens.psl.eu)

THEORETICAL NEUROSCIENCE

TD8: REINFORCEMENT LEARNING

All TD materials will be made available at https://github.com/helene-todd/TheoNeuro2425.

Several learning paradigms exist: mainly supervised, unsupervised and reinforcement learning. In these series of tutorials, we will develop standard models in each of these three paradigms. The third tutorial of these series presents a model of reinforcement learning.

1 The Model

We consider an agent in an environment. The environment can be in any state from a set $\{s\}$ and the agent can perform any action from a set $\{a\}$.

Actions change the state of the world: if at time t, the state of the world is s_t and the agent performs action a_t, then at time $t+1$, the state of the world is s_{t+1} where s_{t+1} is drawn from the probability distribution $p_{tr}(s_{t+1}|s_t, a_t)$.

Actions also allow the agent to obtain reward, depending on which state the world is in: every time the agent performs an action, it receives a reward $r_t = R(s_t, a_t)$.

1. We suppose that the agent can perceive which state s_t the world is in, but does not know the functions p_{tr} and R. How can it learn to choose its actions?

   > Without access to the transition and reward functions, the agent should adopt a model-free strategy consisting in locally learning the best action to perform in each state. This can either be learned with a value-based solution (e.g., Q-learning) or with a policy-based solution (e.g., policy gradient).

2. Given a first policy learned by the agent, how can it avoid sticking to it in case it is suboptimal?

   > This is the **exploration-exploitation** dilemma. To learn what are the best actions to take, the agent has to randomly explore solutions (it is even an obligation at the beginning) in order to infer the best ones from experience. Then it should choose between exploiting the current information it has about the environment (p_{tr} and R) and exploring new solutions.

2 Temporal Difference Algorithm

We suppose that the agent has a policy which allows it to choose what action to perform given the state that it's in. This policy may be stochastic, in which case the action performed at time t is drawn from the distribution $\pi(a_t|s_t)$.

Given this policy, the subjective value of a state $V(s)$ is the expected total reward obtained by the agent supposing that it starts in this state and follows this policy.

3. Express $V(s_t)$ as a function of $V(s_{t+1})$.

> The expected total reward is the sum of the rewards associated with taking given actions a_t, pondered by their probability (given by the policy π), plus the future expected rewards associated with these actions (obtained by summing over the possible states s_{t+1} resulting from taking action a_t, given by p_{tr}):
>
> $$\begin{aligned} V(s_t) &= \sum_{a_t} \pi(a_t|s_t) \left[R(s_t, a_t) + \sum_{s_{t+1}} p_{tr}(s_{t+1}|s_t, a_t) V(s_{t+1}) \right] \\ &= \sum_{a_t} R(s_t, a_t)\pi(a_t|s_t) + \sum_{a_t, s_{t+1}} V(s_{t+1}) p_{tr}(s_{t+1}|s_t, a_t)\pi(a_t, s_t). \end{aligned}$$
>
> Future rewards are generally pondered by a factor γ that we set here equal to 1.

4. Supposing that the agent starts with an estimate $\hat{V}$ of V, suggest an online update of $\hat{V}$ which can be implemented every time the agent performs an action.

> $$\hat{V}(s_t) \leftarrow \hat{V}(s_t) + \epsilon \left[r_{t+1} + \hat{V}(s_{t+1}) - \hat{V}(s_t) \right].$$
>
> The term in the brackets is denoted a Temporal-Difference (TD) error. One can easily see it cancels in average (over the states s_{t+1} resulting from taking action a_t in state s_t) when $\hat{V}$ is equal to V.
>
> The idea of reinforcement learning here is that there exists a function quantifying future rewards in a given state V, and the agent has to learn it, with its estimate $\hat{V}$ in order to take the right actions in given states. However, V is a function of the policy. The policy might be updated as well in order to maximize V.

5. How should the agent modify its policy each time it makes an estimate of V?

> A standard way to implement a policy learning algorithm that allows for fine-tuning the trade-off between exploitation and exploration is the following:
>
> $$\begin{aligned} q(a_t|s_t) &\leftarrow q(a_t|s_t) + \epsilon' \left[r_{t+1} + \hat{V}(s_{t+1}) - \hat{V}(s_t) \right], \\ \pi(a_t|s_t) &= \frac{e^{\beta q(a_t|s_t)}}{\sum_{a'_t} e^{\beta q(a'_t|s_t)}}. \end{aligned}$$
>
> q is an intermediate distribution, which we see evolve with the same updates as $\hat{V}$. The difference is that q is separated into the different actions, such that $q(a_t|s_t)$ is

updated only when action a_t is taken in state s_t, using the TD error, whereas $\hat{V}(s_t)$ is updated when being initially in state s_t, whatever the action taken.

This should ring a bell for those familiar with statistical physics. β plays the role of an inverse temperature. As it goes higher (temperature goes lower), the denominator sum gets dominated by its highest term such that the policy becomes dominated by a single $q(a_t|s_t)$, which is the one maximizing expected future rewards: this is pure exploitation, with the issue that it relies only on what is known up until now about the environment (maybe there are better possible rewards if we a explore a bit?). As it goes lower, all terms in the denominator sum become equivalent and π becomes more and more uniformly distributed: this is pure exploration, with the issue that rewards are then completely random and we never choose the best solution. Fine-tuning β, depending on the task, an optimal trade-off between exploration and exploitation can be found.

3 Maze Learning

We consider a very simple maze: there is a single corridor, the agent starts in the middle and has to find the exit, which is at the right end of the maze. If it reaches the left end of the maze then it must stop.

6. How would you model this?

We model the states of the world as being the 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 'go right' and 'go left'. The reward is 1 at the exit, 0 everywhere else: no reward unless you reach the exit, this is an example of distal reward.

7. Supposing that the agent's policy is to go left or right with equal probability 1/2 at each step, what is the value function?

At any position k not neighbour with n or $-n$, two actions 'go right' and 'go left' are possible, each with probability 1/2 and no reward. Therefore, $V(k) = \frac{1}{2}V(k+1) + \frac{1}{2}V(k-1)$, with the boundary conditions $V(n) = 1$ and $V(-n) = 0$.

$\Delta V(k) = V(k-1) - 2V(k) + V(k+1)$ is the discrete Laplacian, and the previous equation corresponds to $\Delta V(k) = 0$. In the continuous limit this would mean V is an offline function of k ($\partial^2 V = 0 \Rightarrow V(k) = Ak + B$). We take an antatz of this form and use the following boundary conditions:

$$V(-n) = 0 \Rightarrow B = An,$$

$$V(n) = 1 \Rightarrow An + B = 2An = 1 \Rightarrow A = \frac{1}{2n}.$$

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

8. Supposing that the agents initial estimate of V is a uniform function $\hat{V} = 1/2$, how does $\hat{V}$ evolve during the first episode? What is its value at the end of the episode? What is its value at the end of the second episode?

> The agent has no idea where the reward is. All positions look equivalent at first, and we can see it in the TD error $R(k) + \hat{V}(k \pm 1) - \hat{V}(k) = 0$ except at the boundaries. No updates to $\hat{V}$ are made until the agent reaches one of the ends of the maze. If it is the left end, we start again. If it is the right end, we see that at the position $n-1$ an update is made using the value of the reward.
>
> Progressively, with many essays, $\hat{V}$ will be updated from position to position, back-propagating from the right end of the maze.