An Introduction to Reinforcement Learning

Q-Learning

Pei Liu

May 31, 2019

Department of Computer Science @UESTC

Table of contents

Review of basics

Review - MDP

In MDP, the procedure could be simplified as:

- we starts in a state s_0
- we get to choose some action $a_0 \in A$ to take in the MDP
- once the action is executed, the state randomly transitions to some state $s_1 \sim P_{s_0 a_0}$
- we receive the reward $R(s_0)$ or $R(s_0, a_0)$
- $\cdots\cdots$

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \xrightarrow{a_3} \dots$$

Our total payoff is given by

$$R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$$

.

Review - MDP

In MDP, the procedure could be simplified as:

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \xrightarrow{a_3} \dots$$

Our goal in RL is to **choose actions** over time so as to maximize the expected value of the total payoff:

$$R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$$

It is to say that we should find a **policy function** $\pi : S \mapsto A$ mapping from the states to the optimal actions. Under policy π, the total payoff we get is the **value function** as given by

$$V^{\pi}(s) = E\big[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots | s_0 = s, \pi\big]$$

Review - MDP

With the definition of the **value function**

$$V^{\pi}(s) = E\big[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots | s_0 = s, \pi\big]$$

and a fixed **policy** function π. We can know that the value function $V^{\pi}(s)$ satisfies the **Bellman equations**:

$$V^{\pi}(s) = R(s) + \gamma \sum_{s' \in S} P_{s\pi(s)}(s') V^{\pi}(s')$$

which is also called as Dynamic Programming. Bellman's equations can be used to efficiently solve for V^{π}. (i.e. a set of $|S|$ linear equations in $|S|$ variables)

But the solution we want is the optimal value function $V^{*}(s)$.

Example - Flappy Bird

Now, this slide will discuss a RL solution to *Flappy Bird*. And we will focus on Q-Learning algorithm.

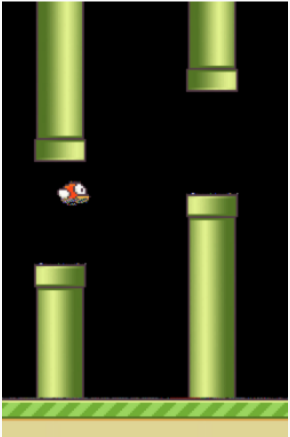

Figure 1: RL based Flippy Bird

Flippy Bird

Description

Only with the current status and reward function, we need to develop an algorithm to let the bird learn to fly.

As talked in MDP, the agent (i.e. clear bird) need to take action w.r.t current status. After receiving the payoff, the agent would update its policy so as to take more optimal action if it is in the same situation.

Some essential concepts:

- status: processed data (distance) or raw data (image)
- actions: click and do-nothing (1 and 2)
- reward: alive = 1, death = -1000, success = 50

Here we talk about a simplifed status - Horizontal and Vertical distance (d_x and d_y).

Q function

Q function (action-utility function) records the payoff of taking an action w.r.t a status.

Table 1: Q function stored in the agent

Status	Fly	Not Fly
(d_{x_1}, d_{y_1})	1	3
(d_{x_1}, d_{y_2})	3	4
...	...	...
$(d_{x_m}, d_{y_{n-1}})$	2	1
(d_{x_m}, d_{y_n})	-100	1

The bird will take an action using the information above. We can take Q function as the brain of bird (or agent).

Algorithm

Q-Learning in Flappy Bird:

```
Initialize Q arbitrarily //随机初始化Q值
Repeat (for each episode): //每一次游戏，从小鸟出生到死亡是一个episode
    Initialize S //小鸟刚开始飞，S为初始位置的状态
    Repeat (for each step of episode):
        根据当前Q和位置S，使用一种策略，得到动作A //这个策略可以是ε-greedy等
        做了动作A，小鸟到达新的位置S'，并获得奖励R //奖励可以是1，50或者-1000
        Q(S,A) ← (1-α)*Q(S,A) + α*[R + γ*maxQ(S',a)] //在Q中更新S
        S ← S'
    until S is terminal //即到小鸟死亡为止
```

Figure 2: Q Learning

Questions?

References

References:

- cnblogs - An introduction to Reinforcement Learning
- DRL of FlappyBird on Github