Report 24/03 : Reinforcement learning for Cache-Friendly Recommendations

Jade Bonnet and Clément Bernard

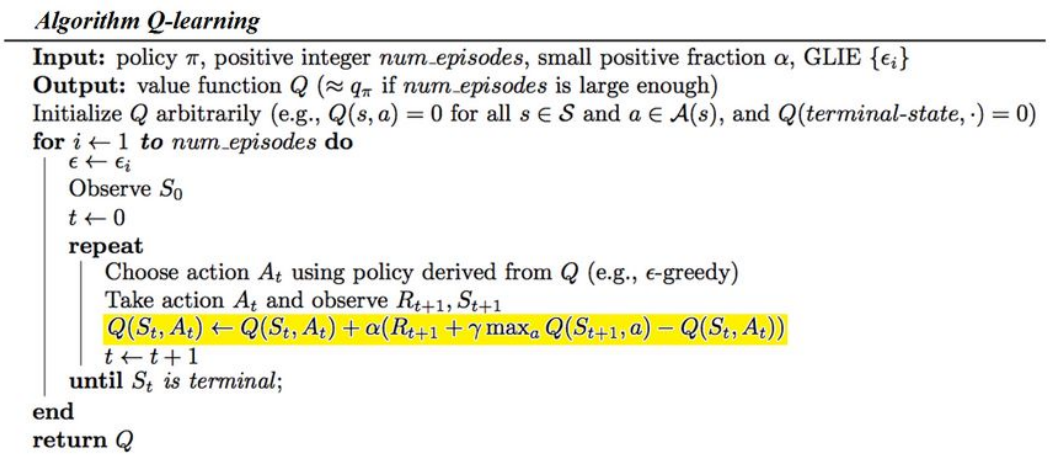
a) Let's focus on the Q(s,a) function: try to write down the Bellman equation for the Q-table updates.

Given a state **s** and an action **a**, we have (Bellman Equation):

$$Q(s,a) \;=\; r \;+\; \gamma \; Max_{a'} \; Q(s',a')$$

It tells us the maximum reward the agent received for entering the current state **s**.
We found the iterative equation :

$$Q_{t+1}(s_t,a_t) \;=\; Q_t(s_t,a_t) \;+\; \alpha(r_{t+1} \;+\; \gamma \; Max_a(\; Q_t(s_{t+1},a\,) - Q_t(s_t,a_t)\,)$$



```
Algorithm Q-learning
Input: policy $\pi$, positive integer $num\_episodes$, small positive fraction $\alpha$, GLIE $\{\epsilon_i\}$
Output: value function $Q$ ($\approx q_\pi$ if $num\_episodes$ is large enough)
Initialize $Q$ arbitrarily (e.g., $Q(s,a) = 0$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$, and $Q(terminal\text{-}state, \cdot) = 0$)
for $i \leftarrow 1$ to $num\_episodes$ do
    $\epsilon \leftarrow \epsilon_i$
    Observe $S_0$
    $t \leftarrow 0$
    repeat
        Choose action $A_t$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
        Take action $A_t$ and observe $R_{t+1}, S_{t+1}$
        $Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1},a) - Q(S_t,A_t))$
        $t \leftarrow t+1$
    until $S_t$ is terminal;
end
return $Q$
```

Figure 1 - Q-learning Algorithm

b) Try to map quantities from the paper to the: actions and states.

1) **E.g., what are the actions of the RL agent (the recommender)? It is the combinations of the N contents to recommend (out of to K). I would suggest to start with the simplest case of N=1.**

i) If N=1, we have only 1 content to recommend. We remind that the agent is the recommended algorithm and the environment is the user.

In this case, as there is only 1 content to recommend, the RL agent can only offer 1 content. The state is therefore either to recommend a cached content or not.
We suppose that the RL agent doesn't control the rest of the N contents of the catalog. (Maybe this supposition is wrong)

ii) For N contents out of K, the RL agent has to recommend N contents. In this case, the immediate action is to offer a combination of N contents with more or less content that is cached or not. It should take into account the other users to see which content is congested. Maybe we can add a coefficient for what is cached or not (which in the paper is denoted as x_i with x_i = 0 for cached-content).

c) User model: the idea is to be able to find solutions for various user models; one user model you could try to "learn" is indeed the markovian one we have on the paper (i.e., click recommendation(s) with fixed alpha or some random content with (1-alpha); what about other user models? E.g. users where the alpha depends on how good the recommended items are?

As we want to add cached contents instead of the classic recommended ones, the constant alpha is not relevant anymore. In this case, maybe we can describe the User Request as follow :

- Probability α_t for the user to pick one of the N recommended contents, with equiprobability $\frac{1}{N}$.
- Probability $1-\alpha_t$ to ignore the recommender and picks any content j from the catalogue with probability p_j .

Maybe we can have an iterative way to describe α_t like :

$$\alpha_{t+1} = \alpha_t + \frac{1}{R_t}\alpha_t$$

or something similar (just to show that the alpha coefficient evolves through the process).
The Markov chain becomes :

$$P_t = \alpha_t Y_t + (1-\alpha_t)P_0$$

d) How will you train this?
We need to have a lot of data to train our model. In fact, we need to provide an environment, which is here a user.

The best case would be to suggest our recommendation algorithm to voluntary users. But as we can’t, we need to modelize multiple users (as each user has different preferences). We can use the previous User Request Model to simulate users and use a dataset of content (like Spotify Dataset).
Then, we can create a simulation with a lot of different users, and we count each content that a user is consuming (from our dataset). We define for each user a number of cached contents. We also can count the number of user that is currently consuming a given content.
In order to modelize the time t, we can create round. For each round, a user consumes a content (which is only for us to select a content or to stop consuming).
Then, we create a loop of training with for each epoch we start another simulation (re initialisation of the users).

What to do next ?