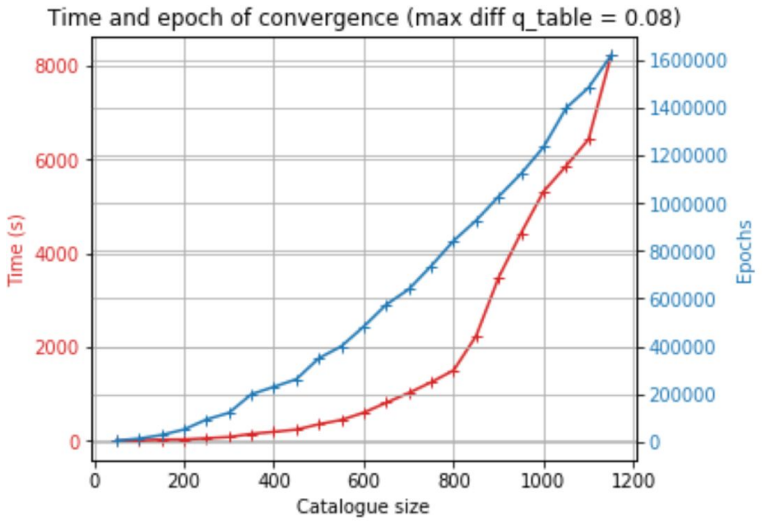
Report 26/05 : Reinforcement learning for Cache-Friendly Recommendations

Clément Bernard and Jade Bonnet

1) Q learning algorithm

a) *Time of convergence*

We updated the number of related and cached contents to be more relevant. We took therefore 5% of the total size of the catalogue related contents and 4% for the cached contents.



Time and epoch to converge for different values of gamma (constraints on related contents)

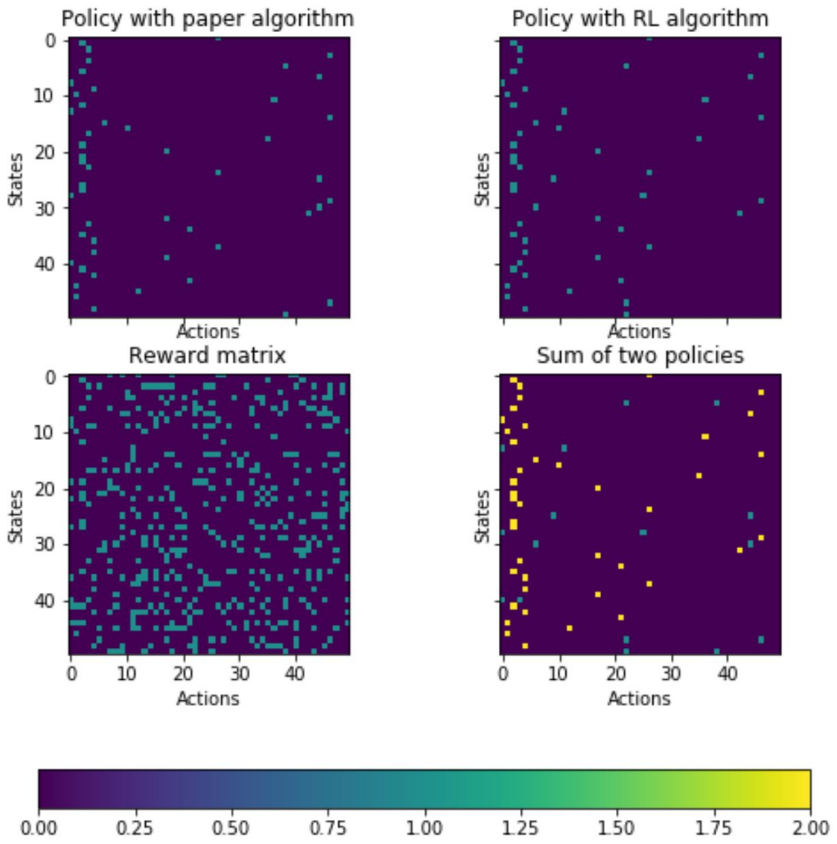
We set the threshold at 0.08. We didn't manage to go more than 1200 for the size of the catalogue.

b) *Comparison with a specific user*

We changed the behaviour of the user as follow :
When the user is in state **i**, he is likely to accept the content **j** that we suggest with the probability : $\alpha_{i,j} = \frac{u_{i,j}}{max_i(u_i)}$

We also made the following rewards :

- Related and cached : 1
- Related and not cached : 0
- Not related and cached : 1
- Neither related nor cached : 0




```
Similarity between Paper algorithm and RL algorithm : 0.84
```

Similarity between paper's policy and algorithm with constraints

We have, for a state which has different recommendation between our algorithm and the paper's algorithm, the q_values as follow :

```
[58]: q_table2[-1]

[58]: array([ 0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
              0.        ,  0.        , 74.16712726, 74.38725067, 74.68235861,
              0.        ,  0.        ,  0.        , 73.33510625,  0.        ,
              0.        ,  0.        ,  0.        , 74.2745436 ,  0.        ,
              0.        ,  0.        , 74.86620473,  0.        ,  0.        ,
             73.99948772,  0.        ,  0.        ,  0.        ,  0.        ,
              0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
              0.        ,  0.        ,  0.        , 74.51555658,  0.        ,
              0.        ,  0.        ,  0.        , 74.01292082,  0.        ,
              0.        ,  0.        ,  0.        ,  0.        ,  0.        ])
```

Q_values for the last state of our algorithm

Here are the parameters we used :

- learning rate =0.2
- gamma=0.99

- epsilon=0.1
- max_iter = 1000000

The final values are pretty close. We think that the max_iter is very large and the algorithm should have reached the optimal policy. Nevertheless, it remains different from the paper, and we don't know why.

2) Deep q learning

a) Pytorch tutorial

We're doing pytorch tutorials to convert our algorithm in pytorch code.

b) Deep q learning pseudo code

We follow this algorithm :

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode $= 1, M$ **do**
  Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
  for $t = 1, T$ **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$
  end for
end for

c) Representation of the states

Our previous mistake was to consider each state differently. Indeed, when we feed the algorithm with [state], it has no knowledge of the environment and so can't infer similarity between two states that should be considered as closed.
If we look to the pseudo code above, what we missed is the $\Phi(s_t)$ function.
Therefore, we thought of this preprocess function :

- $\Phi(S_t) = [1, 1]$ if the state is a content that is related to the previous one and cached
- $\Phi(S_t) = [0, 1]$ if the state is only cached

- $\Phi(S_t) = [1, 0]$ if the state is a content that is related to the previous one

Nevertheless, this representation doesn't take into account the fact that each state has different related contents.
We also thought about this representation :

- $\Phi(S_t = i) = [\ j \ such \ as \ \ u_{i,j} \neq 0\ ]$

We don't know yet if this is a good representation.