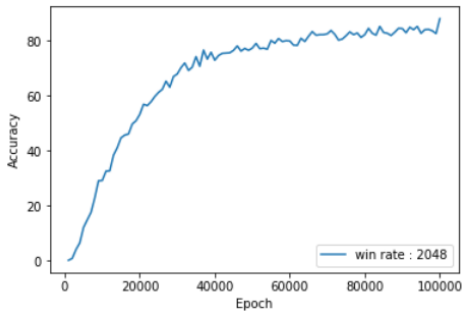
Lab7 : Temporal Difference Learning

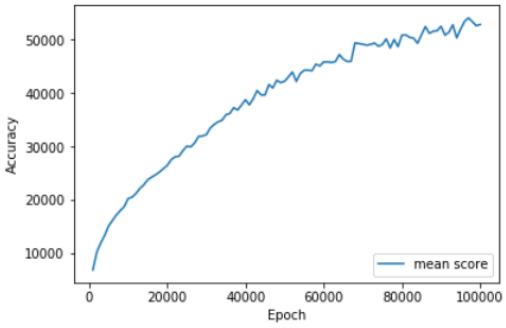
0516069 翁英傑

1. Score Plot :

Win rate of 2048 tile



mean score



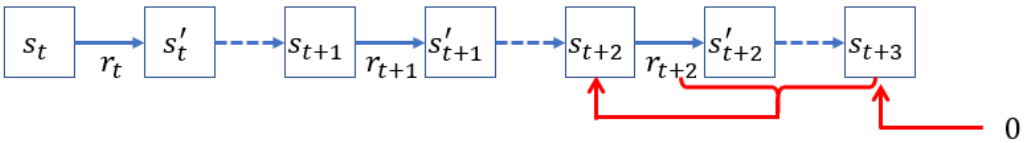
2. Mechanism of TD :

TD learning is to merge the distance of two state to the reward it gets with the action. The formula looks like this:

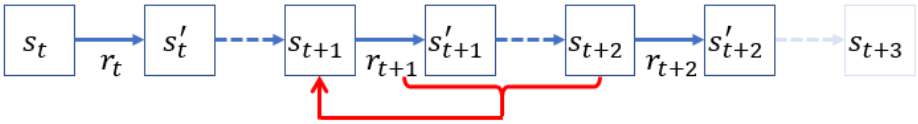
$$V(s) \leftarrow V(s) + \alpha(\overbrace{r + \gamma V(s')}^{\text{The TD target}} - V(s))$$

3. V(state) :

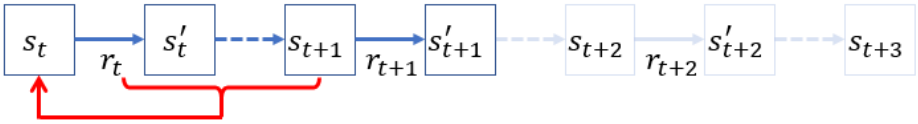
- Step 1: after game over (s_{t+3}), update the last state (s'_{t+2})



- Step 2: update the previous *afterstate* (s'_{t+1})

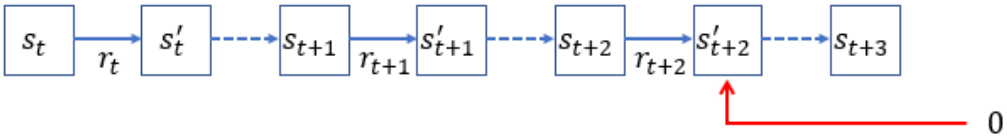


- Step 3: update the previous *afterstate* (s'_t)

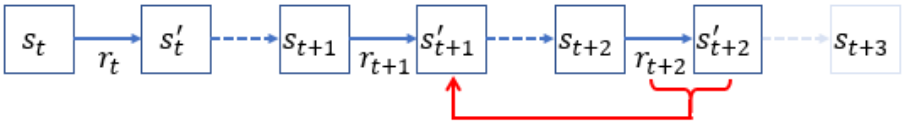


4. V(after-state) :

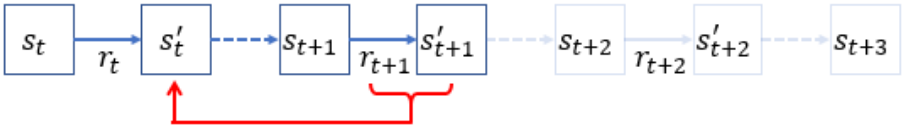
- Step 1: after game over (s_{t+3}), update the last state (s'_{t+2})



- Step 2: update the previous *afterstate* (s'_{t+1})



- Step 3: update the previous *afterstate* (s'_t)



5. Code :

The code is divided into five classes:

- Board: functions of moving the board with up, down, right and left four actions. Also contain the status of the board.
- Pattern: the function for computing the index for storing value of the board

with certain pattern.

- Feature: a virtual class for pattern, has the same function as pattern.
- State: contains the information of each state in a play, which contains what action was taken, what reward was gained, what state turned to what state…etc.
- Learning: main methods of td-learning, including determine function for best action of each state, backward update function…etc.

A pseudocode of a game engine and training (modified backward training method)

```
function PLAY GAME
    score ← 0
    s ← INITIALIZE GAME STATE
    while IS NOT TERMINAL STATE(s) do
        a ← argmax_{a'∈A(s)} EVALUATE(s, a')
        r, s', s'' ← MAKE MOVE(s, a)
        SAVE RECORD(s, a, r, s', s'')
        score ← score + r
        s ← s''
    for (s, a, r, s', s'') FROM TERMINAL DOWNTO INITIAL do
        LEARN EVALUATION(s, a, r, s', s'')
    return score

function MAKE MOVE(s, a)
    s', r ← COMPUTE AFTERSTATE(s, a)
    s'' ← ADD RANDOM TILE(s')
    return (r, s', s'')
```

TD(0)-state

```
function EVALUATE(s, a)
    s', r ← COMPUTE AFTERSTATE(s, a)
    S'' ← ALL POSSIBLE NEXT STATES(s')
    return r + Σ_{s''∈S''} P(s, a, s'')V(s'')

function LEARN EVALUATION(s, a, r, s', s'')
    V(s) ← V(s) + α(r + V(s'') − V(s))
```

TD(0)-afterstate

```
function EVALUATE(s, a)
    s', r ← COMPUTE AFTERSTATE(s, a)
    return r + V(s')

function LEARN EVALUATION(s, a, r, s', s'')
    a_next ← argmax_{a'∈A(s'')} EVALUATE(s'', a')
    s'_next, r_next ← COMPUTE AFTERSTATE(s'', a_next)
    V(s') ← V(s') + α(r_next + V(s'_next) − V(s'))
```