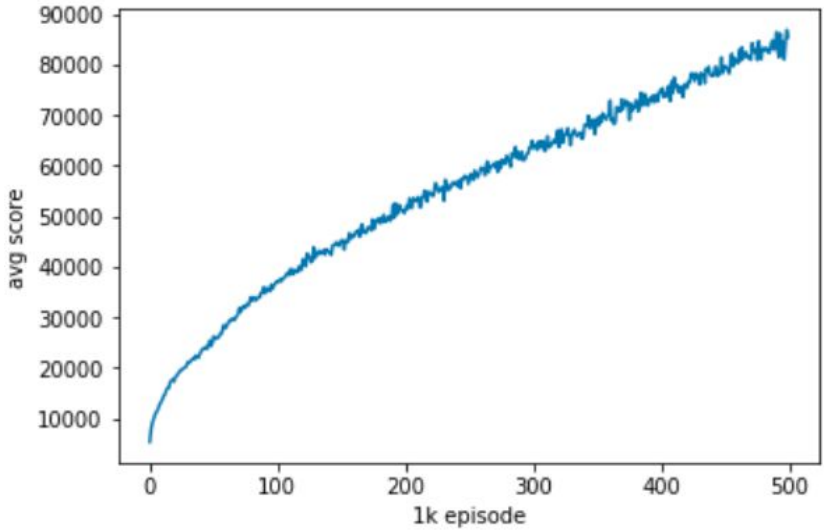
lab7

資工碩一 吳承翰 0856105

1. A plot shows episode scores of at least 100,000 training episodes

共玩500,000次遊戲。



2. Describe your implementation in detail

有五個TODO:

一、

在pattern class中的estimate() function負責計算當前board中的特定patterm(Ex:{0,1,2,3,4,5})與其8個isomorphism Value之和，各個value都是透過查weight table得來的。

```
/**
 * estimate the value of a given board
 * get the weight from this pattern's 8 isomorphic
 */
virtual float estimate(const board& b) const {
    // TODO
    float value=0;
    for(int i=0;i<iso_last;i++){
        value+=(*this)[indexof( patt: isomorphic[i],b)];  // 查 weight table
    }
    return value;
}
```

二、
在TD backup時，我們要去更新4個feature pattern的weight table，float u是已經乘上learning rate的td-error。在此我們更新特定pattern的8個isomorphic feature value，並回傳更新後的值的和。

```
/**
 * update the value of a given board, and return its updated value
 */
virtual float update(const board& b, float u) {
    // TODO
    float value=0;
    for(int i=0;i<iso_last;i++){
        value+=((*this)[indexof( patt: isomorphic[i],b)]+=u);
    }
    return value;
}
```

三、
由於我們在求V(s')時都會去weight table查值，所以我們必須要知道查weight table的門牌號碼，由於是6-tuple network，而且每個tile都用4個bit來表示(可以代表2^0~2^15)，所以門牌號碼是6*4bit=24bit

```
size_t indexof(const std::vector<int>& patt, const board& b) const {
    /**
     * return 6*4=24 bit as index of the weight table
     */
    // TODO
    size_t index=0;
    for(int i=0;i<patt.size();i++){
        index|= b.at(patt[i])<<(i*4);
    }
    return index;
}
```

四、
當要做action時，要選擇最佳的走法使reward+V(s')最大

```
state select_best_move(const board& b) const {
    state after[4] = { 0, 1, 2, 3 }; // up, right, down, left
    state* best = after;
    for (state* move = after; move != after + 4; move++) {
        if (move->assign(b)) {
            // TODO
            // update state's esti = r + V(s')
            move->set_value( v: move->reward()+estimate(move->after_state()));

            if (move->value() > best->value())
                best = move;
        } else {
            move->set_value(-std::numeric_limits<float>::max());
        }
        debug << "test " << *move;
    }
    return *best;
}
```

五、
玩完一個episode後從後面往前依序更新(使用TD-after-state),
td-error為reward[t+1] + V(s'[t+1]) - V(s'[t]), 更新完後把V(s'[t])存起來以供下一次更新使用

```
 * its path would be
 *  { (s0,s0',a0,r0), (s1,s1',a1,r1), (s2,s2,x,-1) }
 *  where (x,x,x,x) means (before state, after state, action, reward)
 */
void update_episode(std::vector<state>& path, float alpha = 0.1) const {
    // TODO
    float exact=0;
    for(path.pop_back();path.size();path.pop_back()){
        state& move=path.back();
        // td_error = r + V(s'[t+1]) - V(s'[t])
        float td_error=move.reward()+exact-move.value();
        exact=move.reward()+update(move.after_state(), u: alpha*td_error);
    }
}
```

3. Describe the implementation and the usage of n-tuple network

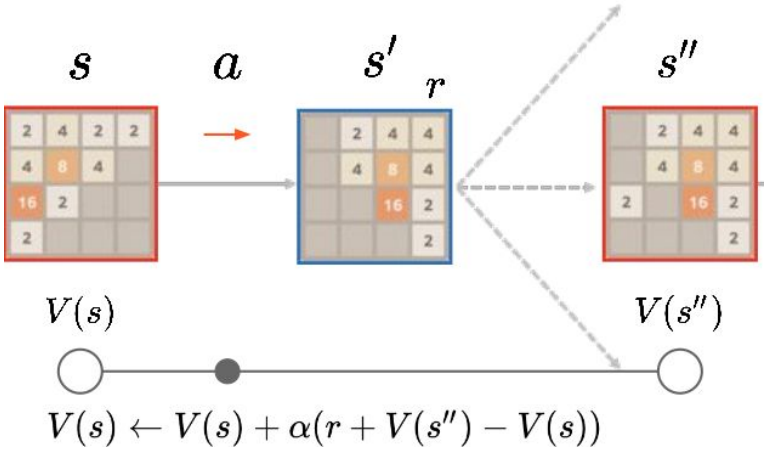
2048的board版面為4*4，人果要紀錄所有state的話個別的V(s)的話，是不可能得，因為state總數共有16^16=2^64個。因此在此使用4 * 6-tuple network，這樣的話只會有4 * 16^6 =2^26種state，大約256MB。一種pattern的8個isomorphism共享同一個weight table。

在feature class中定義了weight指標變數(指向一個去weight table)，這個weight table則用alloac() function來初始化為float[16^6(==16,777,216)]的大小。之後變可以用這個weight table來得知V(s')的值。

```
static float* alloc(size_t num) {
    static size_t total = 0;
    static size_t limit = (1 << 30) / sizeof(float); // 1G memory
    try {
        total += num;
        if (total > limit) throw std::bad_alloc();
        return new float[num]();
    } catch (std::bad_alloc&) {
        error << "memory limit exceeded" << std::endl;
        std::exit( status: -1);
    }
    return nullptr;
}
size_t length;
float* weight;
```

4. Explain the TD-backup diagram of V(state)

V(s)中的s代表的是action後而且經過environment隨機pop out一個tile的狀態，在td-learning的時候，利用剛剛episode紀錄的結果算出td-traget: reward+V(s'')來更新V(s)。



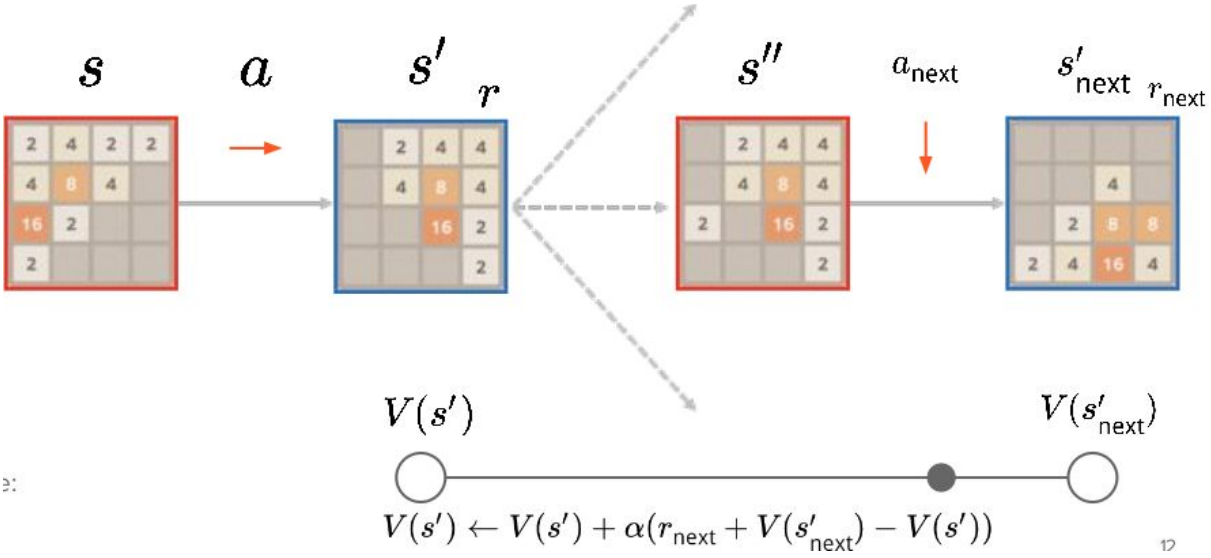
5. Explain the action selection of V(state) in a diagram

在選擇action時，由於after state s' 可以衍生出許多種s''，我們要求出在哪一種action下可以得到最佳的期望值分數: reward+E[V(s'')] (for all possible s'')，所以4種action都要模擬一次並選擇最佳的那個action。

function EVALUATE(s, a)
  $s', r \leftarrow$ COMPUTE AFTERSTATE(s, a)
  $S'' \leftarrow$ ALL POSSIBLE NEXT STATES(s')
  return $r + \Sigma_{s'' \in S''} P(s, a, s'') V(s'')$

6. Explain the TD-backup diagram of V(after-state)

V(s')中的s代表的是剛剛經過action後的狀態，在td-learning的時候，利用剛剛episode紀錄的結果算出td-traget: reward$_{next}$+V(s'$_{next}$)來更新V(s')。



7. Explain the action selection of V(after-state) in a diagram

在選擇action時，由於s會有4種可能的after state s'，我們要求出在哪一種action下可以得到最佳分數: reward+V(s')，所以4種action都要模擬並選擇最佳的那個action。

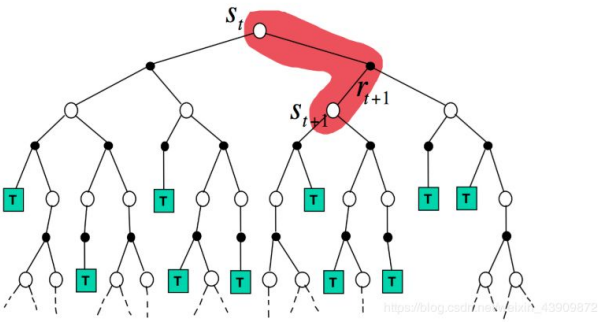
function EVALUATE(s, a)
  $s', r \leftarrow$ COMPUTE AFTERSTATE(s, a)
  return $r + V(s')$

8. Explain the mechanism of temporal difference learning

model-free有2種learning方法，TD-learning與MC-learning，相較於MC-learning，TD-learning可以讓訓練的結果有low variance的特性、但會有bias，也不須等待整個episode跑完就可以透過s_t與s_{t+1}的差異來訓練。

Temporal-Difference Backup

$$V(S_t) \leftarrow V(S_t) + \alpha\left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



9. Explain whether the TD-update perform bootstrapping

TD-learning為bootstrapping，因為它用了estimate value: V(s'next)去update一個estimate value: V(s')

$$V(s') \leftarrow V(s') + \alpha(r_{next} + V(s'_{next}) - V(s'))$$

10. Explain whether your training is on-policy or off-policy

為on-policy，因為behavior policy與target policy相同，都是選擇最佳的action

$$a \leftarrow \underset{a' \in A(s)}{\operatorname{argmax}} \ \text{EVALUATE}(s, a')$$

$$a_{next} \leftarrow \underset{a' \in A(s'')}{\operatorname{argmax}} EVALUATE(s'', a')$$

11. Other discussions or improvements

learning rate如果設太大(Ex:0.1)的話會很容易爆炸，最後使用0.1/32=0.003125作為learning rate，訓練500k次episode。

12. performance

2048的win-rate可以達到94.1%

```
478000  mean = 83300.9  max = 218088
    128 100%    (0.2%)
    256 99.8%   (0.1%)
    512 99.7%   (1.2%)
    1024    98.5%   (4.4%)
    2048    94.1%   (16%)
    4096    78.1%   (49.9%)
    8192    28.2%   (28.1%)
    16384   0.1%    (0.1%)
```