Tugas 3, *Machine Learning*
Laporan Membangun Sebuah Program *Q-Learning* Untuk Menentukan *Optimum Policy* dan Maksimum Total *Reward* Pada Kasus *Grid World*

Oleh:
Enrico Farizky Rustam (1301164263)
IF 40-04 / S1 Informatika / Universitas Telkom

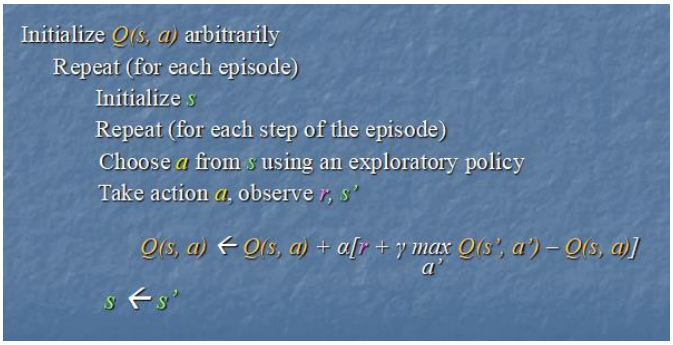
Abstrak: Dalam tugas pemrograman ini terdapat seorang *Agent*. *Agent* diberikan algoritma *Q-Learning* dengan tujuan yaitu, *agent* harus bisa menemukan rute dari *initial state* menuju *goal state*. Algoritma Q-Learning berperan untuk menyimpan *state-state* jalur yang telah dilalui dalam *grid world*. *Agent* hanya bisa melakukan empat aksi: N, E, S, dan W yang secara berurutan menyatakan *North* (ke atas), *East* (ke kanan), *South* (ke bawah), dan *West* (ke kiri). Hal ini akan dilakukan berulang sampai nilai pada *grid world* mencapai nilai yang optimal. Dari hasil uji coba didapatkan data sebanyak 28 state yang telah dipelajari oleh *Agent*. Proses *training* telah mampu meningkatkan pengetahuan *Agent*, sehingga bisa menemukan rute dari *initial state* menuju ke *goal state*. Kemudian dengan Total *Reward* yang didapat sebesar 453.

Kata Kunci : *Q-Learning, Grid World, Agent.*

1. Pendahuluan

Penentuan rute menggunakan Algoritma *Q-Learning* merupakan salah satu algoritma yang digunakan untuk pencarian jalur. Contoh yang dibahas kali ini adalah mengenai pencarian jalur optimal yang dilakukan oleh sebuah *Agent*. *Q-Learning* merupakan salah satu terobosan paling penting dalam *reinforcement learning*.

Berikut Algoritma dari *Q-Learning:*



Berikut Rumus yang digunakan untuk menyelesaikan kasus ini:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a} Q(s',a') - Q(s,a)]$$

Proses pembelajaran *Q-Learning* diawali dengan menginisialisasi nilai *action-value function* Q(S,A) dan proses perulangan pemilihan aksi A serta nilai *action-value function* yang diperbaharui sampai kondisi pembelajaran yang digunakan terpenuhi.

Kelebihan *Q-Learning* adalah sifatnya yang *off policy* (dapat mengikuti aturan apapun untuk menghasilkan aturan optimal), kemudahan algoritma dan kemampuannya untuk konvergen pada aturan optimal. Jadi dengan menggunakan *Q-Learning* akan mampu menyelesaikan permasalahan rute.

2. Deskripsi Soal Masalah

Masalah dari soal ini yaitu membangun sebuah program, dimana file yang diberikan yaitu dataset yang terdapat pada soal.

Kemudian, pada kasus ini dengan menggunakan metode *Q-Learning*, program harus menentukan nilai *optimum policy* sehingga seorang *Agent* yang ada pada kotak start (1,1) mampu menemukan *goal* yang berada posisi (15,15) yang masing-masing dari kotak tersebut berisi nilai acak berupa

bilangan bulat, dengan mendapatkan ***Total Reward*** maksimum pada *grid world*.

Pada kasus ini pula, *Agent* hanya bisa melakukan empat aksi: N, E, S, dan W yang secara berurutan menyatakan *North* (ke atas), *East* (ke kanan), *South* (ke bawah), dan *West* (ke kiri).

3. Metode Penyelesaian

Algoritma *Q-Learning* berperan untuk menyimpan *state-state* jalur *grid world* yang telah dilalui dalam *grid* yang ada. Pada paradigma *Q-Learning*, suatu agent berinteraksi dengan lingkungan dan menjalankan sekumpulan *action*. Lingkungan kemudian dimodifikasi dan agent meng-persepsikan *state* baru melalui sensor.

Pada program *Q-Learning* ini saya menggunakan dataset yang ada untuk membaca data yang dibutuhkan, kemudian menginisiasi beberapa variable sesuai dengan algoritma yang terdapat didalam bab 1.

```
f = open('dataset3.txt', 'r')
x = f.read().split('\n')
f.close()
z = []
for i in x:
    a = i.split('\t')
    z.append(a)
r = 1
gamma = 0.8
min_step = 28 #kemungkinan minimal
untuk mencapai 15x15 (500)
max_step = min_step * 20
eps = 1000
```

Lalu, dilakukan border agar titik bisa terbaca dan menandakan bahwa titik tidak terhubung satu sama yang lain. Dilanjutkan dengan tahap menginisiasi untuk menentukan arah mana yang akan dituju untuk mencapai Goal *state*. Terakhit, memasukkan semua inisiasi kedalam algoritma yang digunakan untuk mencai *goal state*.

```
x_pos = 14 #start x
y_pos = 0 #start y
score = 0
j = 0
print('map : ',map)
while True:
    score += int(z[x_pos][y_pos])
    move =
np.argmax(map[x_pos][y_pos])
    print(x_pos, y_pos) #trace bits
    if(move == 0):
        x_pos  -= 1
    elif(move == 1):
        y_pos += 1
    elif(move == 2):
        x_pos += 1
    elif(move == 3):
        y_pos -= 1
    if(x_pos == 0 and y_pos == 14):
        score += int(z[x_pos][y_pos])
        print('selesai')
        break
    j += 1
    print('gerakan',move)
    print('score',score)
```

4. Output Program

Dalam metode *Q-Learning* ada algoritma untuk menentukan aksi yaitu *control policy*. *Control policy* dalam penelitian sudah berfungsi dengan baik, hal ini terbukti dengan didapatkan tabel yang berisi nilai yang sudah optimal. Keberhasilan proses pembelajaran sebuah *Agent* untuk menemukan rute juga sangat dipengaruhi oleh hasil nilai total *reward* pada tiap *state*.

```
4 11
gerakan 0
score -38
3 11
gerakan 0
score -39
2 11
gerakan 0
score -40
1 11
gerakan 1
score -42
1 12
gerakan 1
score -44
1 13
gerakan 1
score -46
1 14
selesai
```

```
print('final score : ',score)
```

```
final score :  453
```

Output program menghasilkan rute map yang mendefinisikan 4 aksi secara berturut yaitu *North* (ke atas), *East* (ke kanan), *South* (ke bawah), dan *West* (ke kiri) dari masing-masing kurung siku ([ ]). Lalu, terdapat titik kordinat dari hasil yang dicapai untuk sampai ke Goal *state* sebanyak 28 state. Dan yang terakhir, terdapat final score sejumlah 453 yang menjadi hasil Total *Reward* dari kasus Grid World ini.

5. Sumber

1. Available at: **https://piptools.net/algoritma-q-learning/ .** Accessed 27-4-2019, 22:13.
2. Ardiansyah, Ednawati Rainarli. "Implementasi *Q-Learning* dan *Backpropagation* pada Agen yang Memainkan Permainan *Flappy Bird*". Februari, 2017. Available at: http://ejnteti.jteti.ugm.ac.id/index.php/JNTETI/article/viewFile/287/216 . Accessed in 27-04-2019, 22:40.
3. Arifin Samsul, Arya Tandy Hermawan & Yosi Kristian. "Pencarian Rute *Line Follower Mobile Robot* Pada *Maze* Dengan Metode *Q-Learning*". Januari, 2016.