Laporan Tugas Program

Swarm Intelligence: Genetic Algorithm

Disusun Oleh :

Muhammad Naufal Syawali Akbar

1301164488

S1 Teknik Informatika

Telkom University

Bandung

2019

1. Abstraksi

Genetic Algorithm (GA) adalah algoritma yang digunakan untuk mencari genetika yang didasarkan pada ide seleksi alam. GA sangat mengekspoitasi kecerdasan dalam pencarian acak atau random, untuk mendapatkan kinerja kedepan yang lebih baik dengan solusi terbaik, pada umumnya GA digunakan untuk masalah optimasi dan pencarian solusi. Pada kasus ini GA yang dibutuhkan adalah untuk mencari nilai x dan kromosom terbaik.

2. Pendahuluan

GA menjadi salah satu metode yang terdapat dalam bidang kecerdasan kolektif (swarm Intelligence) atau Deep Learning, pada kasus ini diberikan 2 fungsi yang akan menentukan fitness pada GA yang digunakan untuk menentukan X1 dan X2 terbaik dan juga kromosom – kromosom terbaik, dengan metode dekode kromosom, fungsi fitness, pemilihan orangtua dengan turnamen, crossover dan mutasi GA.

3. Analisis

A. Dekode Kromosom

Genetic Algorithm secara dasar sangan bergantung dengan struktur genetik dari kromosom sebelum nantinya akan menjadi populasi maka dari itu langkah pertama adalah dengan dekode kromosom dan menentukan populasi sebelum nantinya di proses dengan skor fitness terbaik. Berikut merupakan hasil running decode kromosome dengan menggunakan numpy random

```
In [8]: # Decode the population
        def decode_chromosome(chromosome):

            c1 = int(chromosome/2)
            #c2 = int(chromosome2/2)
            dc1 = np.zeros(chromosome)
            #dc2 = np.zeros(chromosome2)
            dc1[0: c1] = 1
            #dc2[0: c2] = 2
            np.random.shuffle(dc1)
            #np.random.shuffle(dc2)
            return (dc1)
            #return (dc2)

chromosome = decode_chromosome(20)
#output decode chromosome sebanyak 20
print ("Hasil Chromosome: \n", chromosome)
```

```
Hasil Chromosome:
 [0. 0. 1. 1. 1. 0. 0. 0. 1. 1. 0. 0. 0. 1. 1. 1. 0. 1. 0. 1.]
```

B. Fungsi Fitness

Fungsi fitness menjadi salah satu cara untuk menentukan kromosom, individu, dan population terbaik, dalam kasus ini diberikan 2 fingsi fitness yaitu

$$f(x_1, x_2) = -\sum_{i=1}^{5} i\,cos((i+1)x_1 + 1) \sum_{i=1}^{5} i\,cos((i+1)x_2 + 1)$$

dan

$$f(x_1, x_2) = -cos(x_1)cos(x_2)exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$$

Berikut adalah hasil dengan menggunakan 2 fungsi fitness diatas dengan nilai terbaiknya 10

```
def func1(x1,x2):
    f1 = 0
    f2 = 0
    for i in range(1,6):
        f1 += i*np.cos((i+1)*x1+1)
        f2 += i*np.cos((i+1)*x2+1)

    f3 = f1*f2*-1
    return(f3)
```

```
def func2(x1,x2):
    #fungsi 2
    fu2 = -np.cos(x1)*np.cos(x2)*np.exp(-(x1-np.pi)**2-(x2-np.pi)**2)
    return(fu2)

func2(3.14,3.14)
```

```
def fitness(func, x1, x2):
    #pilih nilai c bebas asal diatas 1
    c = 1.2

    #variabel nyimpen func
    h = func(x1,x2)
    return(c**h)
```

```
#fitness_score1 menggunakan function fitness 1 lanjutin yang ini
fitness_score1 = fitness(func1, chromosome, population)
print ("\n fitness terbaik dari function 1: \n",fitness_score1)

#fitness_score2 menggunakan function fitness
fitness_score2 = fitness(func2, chromosome, population)
print ("\n fitness terbaik dari function 2: \n",fitness_score2)
```

```
identik atau terbaik:
[10  9 10 11]

fitness terbaik dari function 1:
[[1.10779277e-04 1.10779277e-04 9.80878354e-04
 9.80878354e-04 1.10779277e-04 1.10779277e-04 !
```

```
fitness terbaik dari function 2:
[[0.99999995 0.99999995 0.99999447
 0.99999995 0.99999447 0.99999447 (
 0.99999995 0.99999447 0.99999447 (
```

C. Pemilihan Orang tua

Pemilihan orang tua menggunakan metode turnamen, sebagai berikut

```
In [14]: #memilih parent dengan cara turnament breed dengan menggunakan metode atau procedur kromosom dan population
def turney_individu(population, best):
    #dapatkan terlebih dahulu ukuran dari populasinya
    pop_size = len(best) #len mengambil string sebagai parameter

    #pilih individu untuk menjadi anggota turnamen
    in1 = random.randint(0, pop_size-1)
    in2 = random.randint(0, pop_size-1)

    #score masing masing individu
    sc_ind1 = best[in1]
    sc_ind2 = best[in2]

    #bandingkan dan pilih yang terbaik
    if sc_ind1 >= sc_ind2:
        best_ind = in1
    else:
        best_ind = in2

    #return individu terbaik untuk dijadikan parent nantinya
    return population[best_ind, :]
```

```
# setelah mendapat individu terbaik dari turnamen

# waktunya untuk test seleksi orang tua

chromosome = decode_chromosome(20)
population = create_population(4,20)
best = best_pop(chromosome, population)

#tampilkan nilai dari kedua orang tua
ortu1 = turney_individu(population, best)
ortu2 = turney_individu(population, best)
print ("Chromosome Orang tua:")
print (ortu1)
print (ortu2)
```

```
Chromosome Orang tua:
[0. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1.]
[1. 0. 0. 0. 1. 0. 0. 1. 1. 1. 1. 0. 1. 0. 1. 0. 1. 1. 0. 0.]
```

D. Metode Crossover

Metode Crossover menggunakan 2 kromsom orang tua yang sudah di turnamenkan, dengan menghasilkan 2 anak sebagai berikut

```
Chromosome Ortu yang dipakai:

[1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1.]
[0. 0. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0.]

 Hasil anak dari crossover:

[1. 0. 1. 1. 1. 0. 0. 1. 0. 0.]
[0. 0. 1. 0. 0. 1. 0. 1. 1. 1.]
```

E. Metode Mutasi

Probabilitis pada metode mutasi penulis memasukkan dengan nilai 0.1

GA akan berhenti setelah melakukan 5 kali loop sesuai dengan fitnessnya