EVOLUTIONARY ALGORITHM

Sharjeel Akram
BSCS-2019-60

October 31, 2021

ARTIFICIAL INTELLIGENCE
CS-340
Dr. Junaid Akhtar

Acknowledgement

I would like to express the deepest gratitude to Professor Dr. Junaid Akhtar, my teacher, for his help, guidance and encouragement in my work and far beyond. Without his numerous suggestions and immense knowledge, this would never have been completed. Without his open-mindedness, my hard-work would not have been so rewarding.
I appreciate the many inspiring discussions with Ali Raza Khan, Student of 4th year Computer Science Department, T.A to Dr Juniad, who gave me the eye-opening opportunity to try out this evolutionary algorithm.
It gives me great pleasure to have the opportunity to thank my class fellows who helped me in developing the algorithm at Namal University Mianwali.

Table of Contents

List of Figures

Abstract

The evolutionary algorithm is a adaptable approach for settling search and optimization issues. It depends on experience living beings' genetic process. All through ages, population advance in nature as indicated by Darwin's theory of natural selection and survival of fittest. Evolutionary Algorithm can create solutions for real-world issues by replicating this interaction. The evolution of these solutions towards ideal issues, is vigorously dependent on their legitimate coding. An evolutionary algorithm is a mathematical function or programming software that inputs specimens as data sources and yields which ones should create posterity for the next generation. Further advanced evolutionary algorithm can give an iterative cycle wherein the species is straight forwardly taken and a new generation is made, which replaces the past one various occasions, chose by its own plan. One of its significant characteristics is that it idealizes its own heuristic all through the execution interaction, keeping away from the requirement for broad times of specific human training.

1 Problem Statement:

In the course of Artificial Intelligence, we had to find a template image of a "boothi" in the group image of different people. We have to develop an algorithm which should be generalized and can be implemented for all types of this problem. The most important thing about algorithm was that it should be implemented on the basis of theory of natural selection and evolution by Charles Darwin.

2 Introduction:

The use of nature-inspired algorithms has grown in popularity throughout the most recent couple of years. The study of natural processes turned into foundation for nature-inspired algorithms. Diversification (exploration) and escalation (exploitation) are two typical essential provisions in nature-inspired algorithms. Exploration is the quest for worldwide optima through the random exploration of new solution spaces, while exploitation is the chase after nearby optima in currently concentrated on arrangement spaces. Intensive exploration does not yield an optimal solution, while extensive exploitation traps the algorithm in local optima. Therefore, every nature-inspired algorithm should find some kind of harmony between the two.

2.1 Darwin's Theory of Evolution:

The Evolutionary Algorithm depends on the natural selection hypothesis' idea of Charles Darwin. As indicated by Darwin's theory of evolution, all life is associated and plunges from a common predecessor. The theory suggests that refined organic entities grew naturally through time from less complex ancestors. Basically, when random genetic mutations emerge in a creature's hereditary code, the positive changes are kept in light of the fact that they assist the living being survive, a process known as "natural selection". Charles Darwin at initially presented natural selection as a natural theory. These profitable mutations are given on to individuals in the new generations. Beneficial mutations accumulate all through time, coming about in a totally new creature. Darwin's theory of evolution, i.e., natural selection, is the establishment of the Evolutionary algorithm, which reproduces the fittest qualities.
In nature, the individual of a population compete with one another in the

quest for assets like food, water and haven. Indeed, even individuals from similar species frequently compete in the quest for a partner. Those individuals who are more fruitful in surviving and attracting partners are bound to produce an enormous number of posterity. On the other hand, less talented individuals will deliver fewer descendants. This implies that the qualities of the best adapted individuals will spread in progressive generations to a growing number of individuals. The combination of good attributes from various ancestors can now and again create "super individual" descendants, whose adaptation is a lot more prominent than that of any of their predecessors. In this way, attributes progressively better adapted to the climate wherein they live.

Evolutionary Algorithm utilizes an immediate similarity with natural behaviour of Darwin's theory. In nature, every individual is assigned a worth or score, identified with the goodness of said solution. This would be comparable to the level of effectiveness of an organic entity to go after specific resources. The more prominent the adaption of an individual to the issue, the more probability, he will be chosen to reproduce, crossing his hereditary material with one more individual chose similarly. This crossing will create new generations, descendants of the past ones, which share a portion of the characteristics of their ancestors.

The smaller the adaption of an individual, in this way, another population of potential solutions is created, which replaces the past one and confirms the fascinating property that contains a more extent of good attributes contrasted with the past population. Consequently all through the generations, great qualities spread through the population. By preferring the crossing of the best adapted individuals, the most encouraging areas of the search space are being investigated. On the off chance that the Evolutionary Algorithm has been all around planned, the population will join towards an optimal solution of the issue.

2.2 What is Evolutionary Algorithm?

In recent years, there has been a uproar comparable to Artificial Intelligence (AI). Huge organizations like Google, Apple and Microsoft are effectively dealing with this issue. In fact, AI is only an umbrella that covers numerous objectives, approaches, apparatuses and applications. The evolutionary algorithm is a search system dependent on Darwin's theory of evolution. The evolutionary algorithm is one of the most generally utilized algorithms, with

applications optimization, design, and application domain. The thought is that "evolution" tracks down an optimal solution for the problem after various progressive ages like normal determination.
The evolutionary algorithm imitates three evolutionary processes: selection, crossover and mutation. Its algorithm is population based.To evaluate the fitness of each individual in the population, a fitness function is used. The best solutions are picked at random utilizing a selection function to further develop inferior solutions.Evolutionary Algorithm is a generally famous search and optimization method for resolving highly intricate problems.
The success of this algorithm has been demonstrated in regions involving machine learning approaches.. This works on a population of potential solutions applying the principles of natural selection to deliver better and better approximations to a solution. At each generation, another arrangement of solutions is made by the most common way of selecting individuals as per their degree of fitness in the problem domain and reproducing them together utilizing administrators acquired from natural hereditary qualities. This interaction prompts the evolution of population of individuals that are more qualified to their current circumstance than the individuals that they were made from, similarly as in natural adaption.
In Evolutionary algorithm, arrangements are para-metrically addressed in strings of code (for example binary in the form of 0's and 1's). Fitness value is characterized to assess arrangements. The overall methodology of a evolutionary Algorithm include:

1. Create a population of irregular individuals

2. Evaluate every individual's fitness

3. Select individuals to be parents

4. Select individuals to be parents

5. Produce children

6. Evaluate children

7. Repeat stages 3 to 5 until an answer with fulfilled wellness is found or some foreordained number of ages is met.

2.2.1 Evolutionary Algorithm as Meta-heuristic Algorithm:

Meta-heuristic algorithms are amazing and effective apparatuses to settle optimization solutions. Meta-heuristic algorithms are officially characterized as iterative age processes which guide a subordinate heuristic by consolidating wisely unique ideas for investigating and taking advantage of the inquiry space. Learning techniques are utilized to structure data to find effectively approach ideal arrangements.[2]

3 How Darwin's Theory is related to Evolutionary Algorithm?

3.1 Population Initialization:

The origin of species depends on "Protection of favorable variations and dismissal of unfavorable variations ". The variation refers to the changes shown by the individual of a species and furthermore by posterity of similar ancestors. There are a lot of individuals brought into the world than can survive, so there is a nonstop struggle forever. Individuals with a benefit have a more opportunity for survive i.e.,survival of fittest (based on Darwin's theory of natural selection). For instance, Giraffe with long necks can have food from tall trees too from grounds, then again goat, deer with little neck have food just from grounds. Accordingly, regular choice assumes a significant part in this surviving cycle. On the comparable line in an Evolutionary algorithm, in every cycle the fittest individual are survived and the non-fittest individuals are died. In every iteration the process continues forever until a phase came to in which the stable or optimized solutions are reached, which can be named as adaptability.

```
def initialization_population(rows, columns,pop_size):
    random_points1 = np.random.randint(rows, size=pop_size)
    random_points2 = np.random.randint(columns, size=pop_size)
    a_zip = list(zip(random_points1,random_points2))
    return a_zip
```

Figure 1: Initialization of Population of function

3.2 Selection of Fittest Ones:

The process of selection serves to pick the most adjusted individual of the population, with the goal that they go about as ancestors of the next generation. In nature there are a few factors that intercede so an individual can have posterity. The as a matter of first importance is that it figures out how to endure, either on the grounds that it isn't eaten by hunters, or on the grounds that it can secure food. Interestingly, you find a partner to reproduce The last factor is that the blend of the two people is adept to make another individual.
In evolutionary algorithm, selection is a bunch of decides that serves to pick the guardians of the next generation. These guardians will repeat (hybrid) and create posterity. A framework generally utilized in evolutionary algorithm is the selection per every population. This framework comprises in haphazardly browsing the populace a specific number of people.

```python
def fitness_evaluation(image, boothi,random_points):
    rows = []
    columns = []
    fitness=[]
    for x,y in random_points:
        rows.append(x)
        columns.append(y)
    for i in range(len(random_points)):
        if rows[i] + 35 < 512 and columns[i] + 29 < 1024:
            num = image[rows[i]:rows[i]+35,columns[i]:columns[i]+29]
            co_relation = np.corrcoef(boothi.ravel(),num.ravel())
            fitness.append(co_relation[0,1])
        else:
            fitness.append(-1)
    return fitness


def Selection(random_points,fitness):
    Ranked_Pop = []
    Ranked_Pop = list(zip(random_points,fitness))
    Ranked_Pop.sort(key=lambda x:x[1],reverse=True)
    Rand_points = []
    Fitness = []

    for x,y in Ranked_Pop:
        Rand_points.append(x)
        Fitness.append(y)
    return Rand_points, Ranked_Pop
```

Figure 2: Selection of fittest Ones

3.3 Crossover:

During this stage, the individuals which were chosen in the selection stage, are crossed or blended. That is, the qualities of the two parents are combined as one to bring about the various children. There are a few techniques for intersection, however which i used in my evolutionary Algorithm is stated below:

1. Crossover based on a point: the two individuals chose to assume the part of parents, are recombined through the determination of a cut point, to later exchange the sections that are to right said point. That is, the qualities of the first on the left of the cut-off point are recombined with the qualities of the second on the right of the cut-off point and the qualities of the first on the right of the cut-off point are recombined with the qualities of the second on the left of the cut-off point.

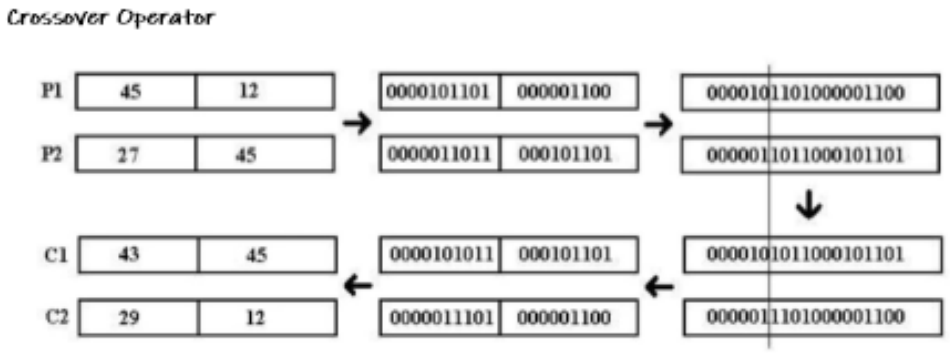

Figure 3: This shows the crossover in Evolutionary Algorithm

```
def Cross_Over(Rand_points):
    a = []
    b = []
    a_bin = []
    b_bin = []
    for x,y in Rand_points:
        a.append(x)
        b.append(y)
    for i in range(len(a)):

        a_bin.append(bin(a[i])[2:].zfill(9))
        b_bin.append(bin(b[i])[2:].zfill(10))
    A_list =[]
    for i in range(len(a_bin)):
        A_list.append(a_bin[i] + b_bin[i])
    New_generation = []
    for i in range(0,len(A_list),2):

        slicer = np.random.randint(0,18)
        left1 = A_list[i]
        left_left = left1[:slicer]
        left_right = left1[slicer:]
        left2 = A_list[i+1]
        left2_left = left2[:slicer]
        left2_right = left2[slicer:]
        temp1 = left_left + left2_right
        temp2 = left2_left + left_right
        left_temp1 = temp1[:9]
        right_temp1 = temp1[9:]

        left_temp1 = int(left_temp1,2)
        right_temp1 = int(right_temp1,2)
        New_generation.append((left_temp1,right_temp1))
        left_temp2 = temp2[:9]
        right_temp2 = temp2[9:]
        left_temp2 = int(left_temp2,2)
        right_temp2 = int(right_temp2,2)
        New_generation.append((left_temp2,right_temp2))
    return New_generation
```

Figure 4: Crossover in Algorithm

3.4 Mutation:

The mutation is considered as a fundamental operator, which gives a small element of randomness in the individuals of the population. In spite of the fact that it is acknowledged that the crossover operator is responsible for searching the space for optimized solutions, the mutation operator is responsible for expanding or lessening the search space inside the evolutionary algorithm and the advancement of variability, There are a few methods to apply the mutation to individuals in a population but the method for mutation which i used in my evolutionary algorithm is stated below:,

1. Mutation based on a point: the two individuals chose to assume the part of parents, and then there is randomly cut-off point on them. At the random cut-off point, if there is value of 0, it is replaced by 1 and if it is 1, it is replaced by 0. Then it is again recombined to turn themselves into individuals.

Mutation

Mutation is a unary operator that is applied to a single individual with a very small mutation probability. All it does is to flip a bit from 0 to 1, or 1 to 0, and hence produces a variation in the population. Together with crossover, it gives exploratory powers to an EA.

Figure 5: This shows the mutation in Evolutionary Algorithm

```
def Mutation(Evolved_Pop):
    a = []
    b = []
    a_bin = []
    b_bin = []
    for x,y in Evolved_Pop:
        a.append(x)
        b.append(y)
    for i in range(len(a)):

        a_bin.append(bin(a[i])[2:].zfill(9))
        b_bin.append(bin(b[i])[2:].zfill(10))
    A_list =[]
    new_list = []
    for i in range(len(a_bin)):
        A_list.append(a_bin[i] + b_bin[i])
    for i in range(len(A_list)):
        random = np.random.randint(0,19)
        Check = list(A_list[i])
        if Check[random] == '0':
            Check[random] = '1'
            temp1 = Check
            temp1 = ''.join([str(elem) for elem in temp1])
            left_temp1 = temp1[:9]
            right_temp1 = temp1[9:]
            left_temp1 = int(left_temp1,2)
            right_temp1 = int(right_temp1,2)
            new_list.append((left_temp1,right_temp1))
        elif Check[random] == '1':
            Check[random] = '0'
            temp2 = Check
            temp2 = ''.join([str(elem) for elem in temp2])
            left_temp2 = temp2[:9]
            right_temp2 = temp2[9:]

            left_temp2 = int(left_temp2,2)
            right_temp2 = int(right_temp2,2)
            new_list.append((left_temp2,right_temp2))
    return new_list
```

Figure 6: Mutation Function in Algorithm

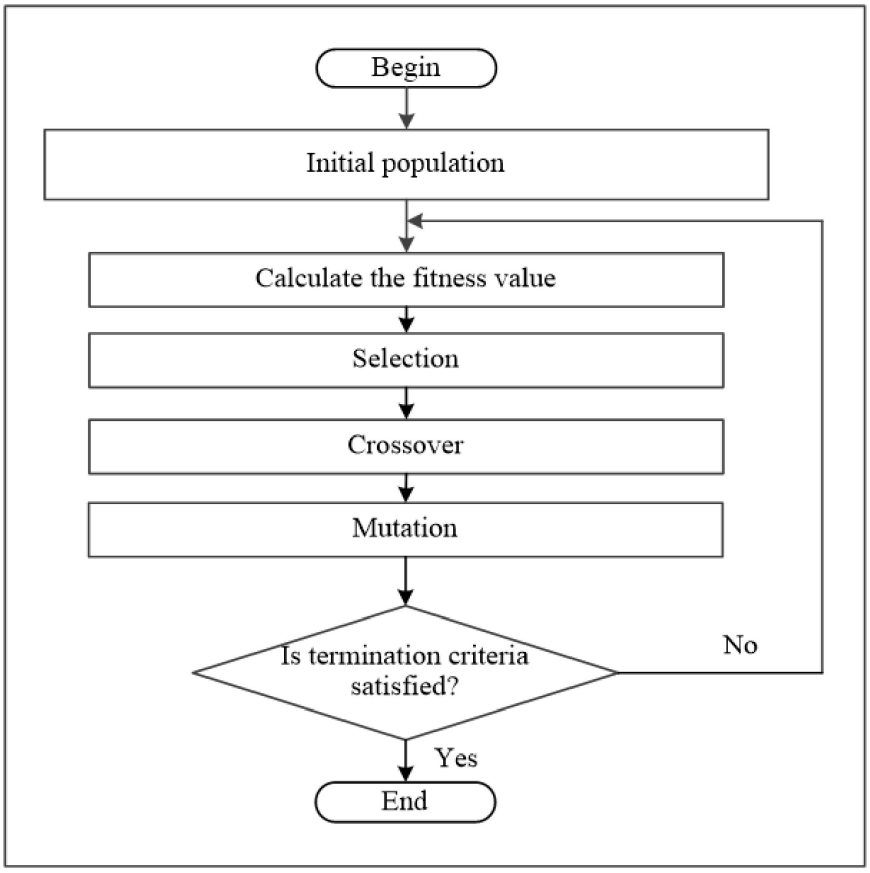

Figure 7: This shows the flow chart of Evolutionary Algorithm

4 Experiments and Results:

4.1 Experiment No 1:

4.1.1 Hypothesis:

My evolutionary Algorithm is basically to find a template image in a group image using the concepts of Darwin's theory of evolution. My first hypothesis was that if I just pick 1000 random points from the group Image, I will be able to find the the template image in the group image.

4.1.2 Testing:

When I started my evolutionary algorithm without doing crossover and mutation, it did not even showed the result of template image even after 5000 random points.

4.1.3 Results:

My hypothesis went wrong because what I thought that I may get template image point after drawing 5000 random points on the group image but the below given graph is a straight line graph, showing nothing special as a ranked population of 5000 points.

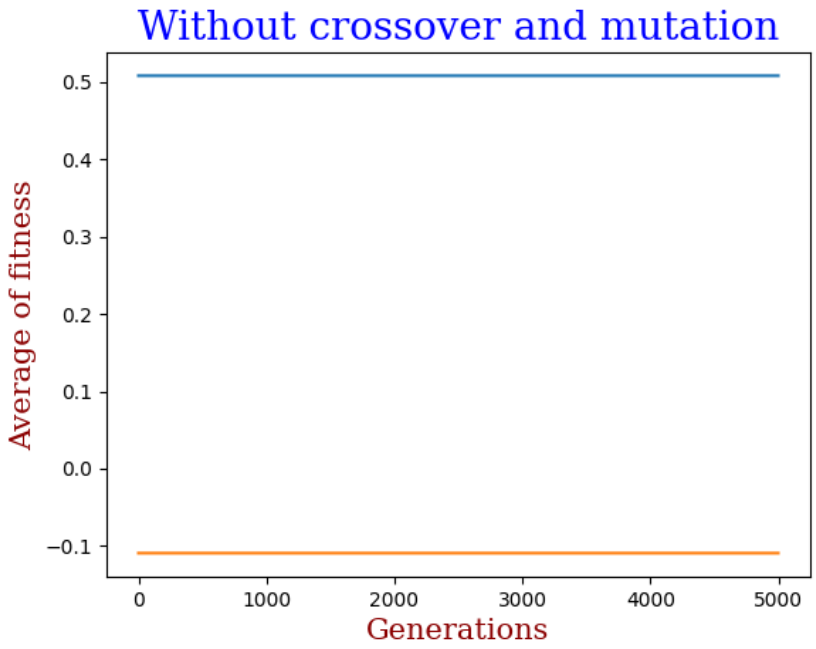

Figure 8: This shows the generation without crossover and mutation.

4.2 Experiment No 2:

What will happen, if I do crossover and mutation?

4.2.1 Hypothesis:

I did not get any point or template image in group image after the selection of 5000 random points on group image. I thought that I should go for crossover and mutation function based on the Darwin's theory of evolution. I may get the template image as well as a linear graph of average and mean fitness value of each generation.

4.2.2 Testing:

I started my evolutionary algorithm and and built a function of crossover and mutation separately. I gave the the output of selection function which was ranked population as an input to the crossover function and output of crossover function to the mutation function as an input. I ran this algorithm with initially 100 random points of population till 5000 generations.

4.2.3 Results:

As a result of crossover and mutation function in evolutionary algorithm, I was able to get the template image points in the group image but my one hypothesis also went wrong at this time that I could not get the linearly graph. Here is the graph of 100 points till 5000 generations but got the point only in 39 generations.

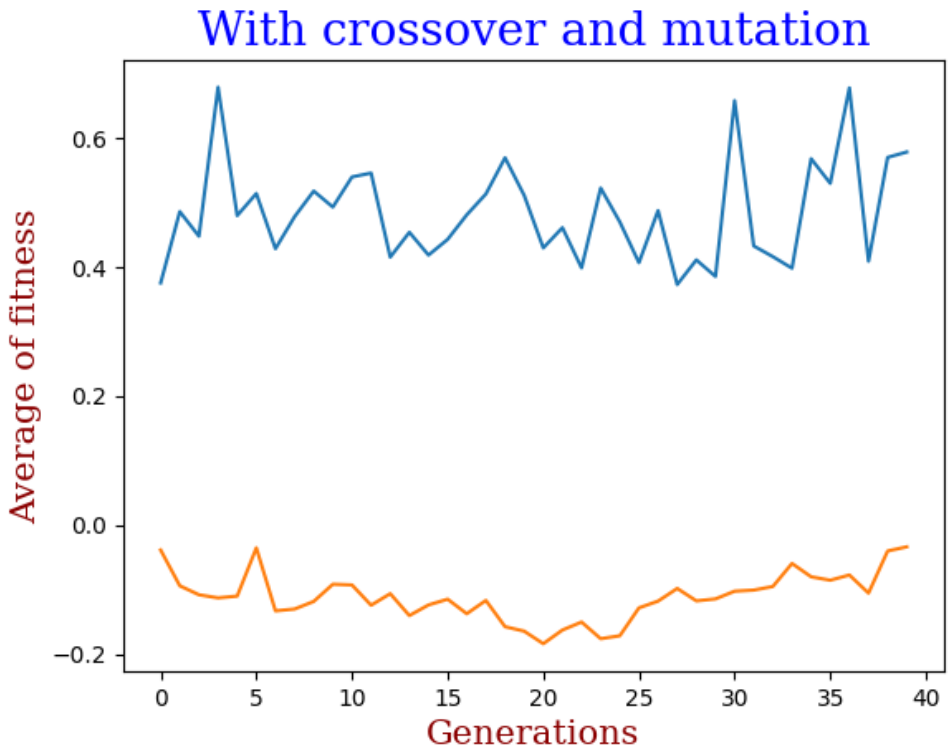

Figure 9: This shows the generation with crossover and mutation.

Figure 10: This shows template image in group image.

4.3 Experiment No 3:

What will happen, if I close the function of mutation and evolve generation on the crossover function?

4.3.1 Hypothesis:

I was not getting the linear graph of max fittest point and mean of fitness value of each generation, so I closed the mutation function and thought that I may get the linear graph of max fittest point and mean of fitness value of each generation.

4.3.2 Testing:

I started my evolutionary algorithm and closed the function of mutation and gave the Evolved Pop to the fitness function as an input and started finding the template image and linearly graph.

4.3.3 Results:

When the code started working, it took maximum time in crossover function and sometimes after maximum number of population it even did not return any result, neither the template in image nor the linear graph. Sometimes it gives me result in only 7 generations and sometimes it gave me result after 187 generations because of random initialization of population but the graph was not linear.

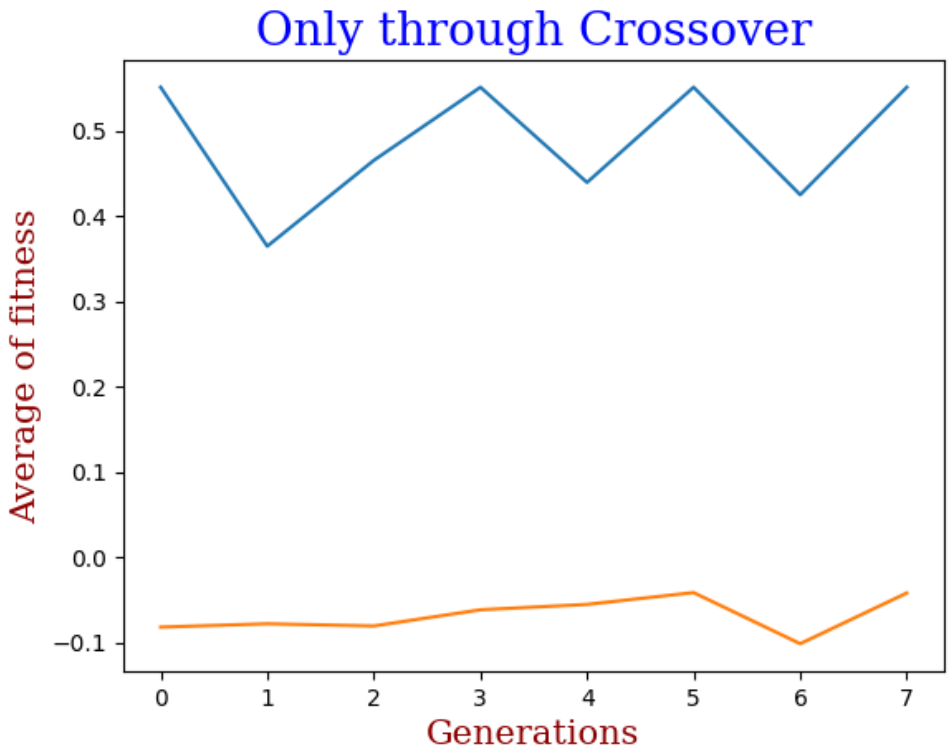

Figure 11: This graph shows without mutation function after 7 generations

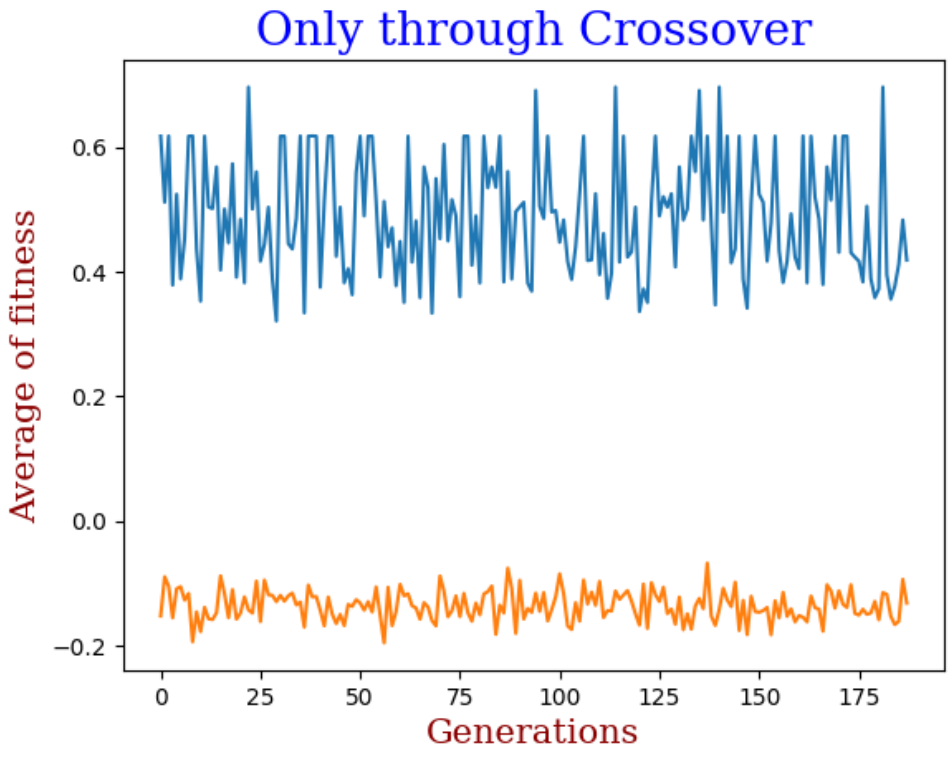

Figure 12: This graph shows without mutation function after 187 generations

4.4 Experiment No 4:

What will happen, if I increase random population size and generation gap?

4.4.1 Hypothesis:

I was trying to get linear graph of average fittest and mean fittest point of each generation and tried many experiments but did not get the better result. I thought that If I increase the random population size till 5000 and generations till 1000, I may get the linear graph of average fittest and mean fittest point of each generation.

4.4.2 Testing:

I started my evolutionary algorithm and increase the population size till 5000 and reduced the generations value from 5000 to 1000. Now I started finding template image in group image and linear graph for which I was doing all experiments.

4.4.3 Results:

When the code started working, it took maximum time in crossover function and mutation function because this time random population was increased 50 times more than before.I found the template image in group image but graph was not linear.

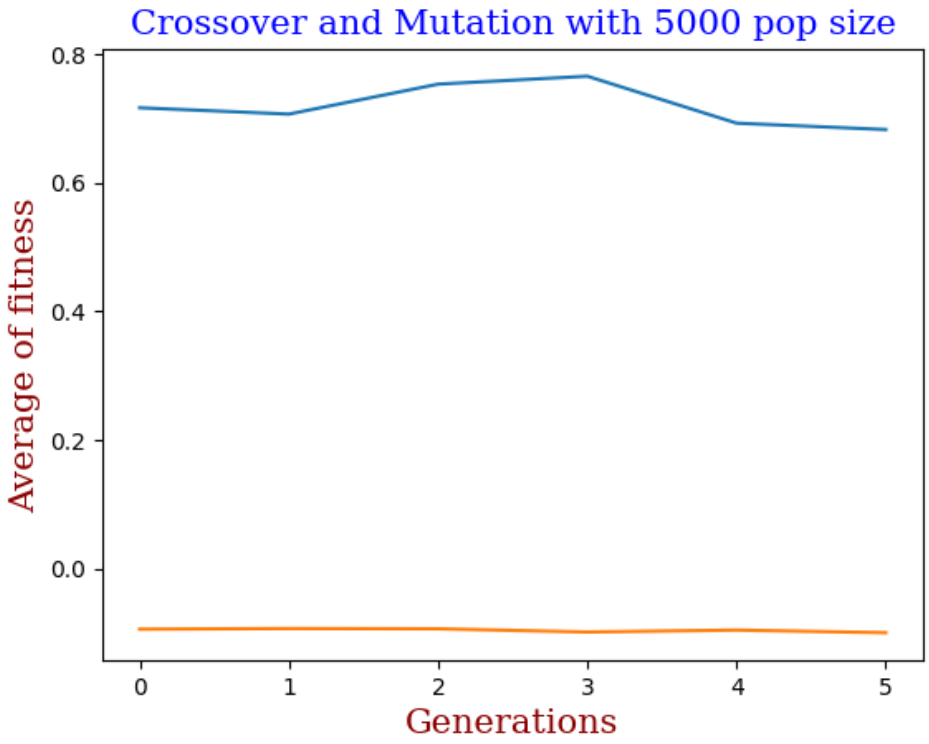

Figure 13: This graph shows with 5000 generations and 5000 pop size

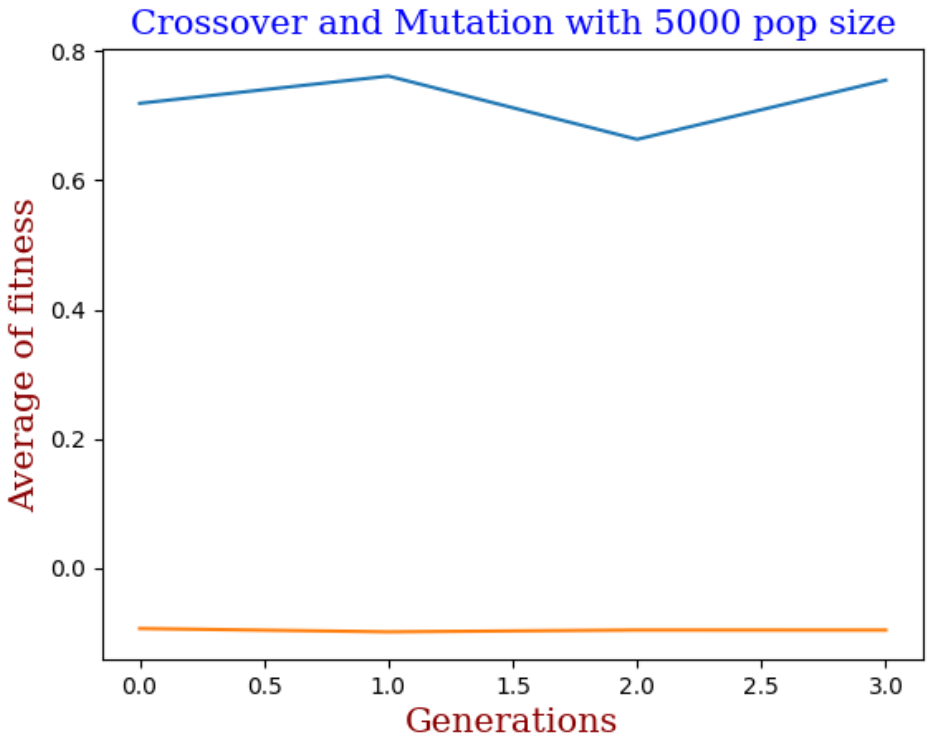

Figure 14: This graph shows with 5 generations and 5000 pop size

5 Limitations of Evolutionary Algorithm

Despite its proven capacity in resolving numerous search and optimization problems, The Evolutionary Algorithm still has several weaknesses:

1. The new population is generated from the existing parents.
2. The randomness involvement that happened in the population at the time of solution initialization may both lead to making the evolutionary algorithm fail to create exploration in the solution search space.
3. In addition, the mutation is applied to the new chromosomes (offspring) that are obtained from the crossover operation and then moved to the next generation (using same pool) that might lead to not providing a good enough exploration and solution diversity. In other words, the crossover followed by mutation may prevent the mutation and crossover operations from guaranteeing an optimal solution.[1]

My Evolutionary Algorithm is now capable of finding any template image in the bigger size image. I tried my best to make it efficient algorithm by plotting the linearly graph but I could not plot. I even changed many things in the code. For example, first of all I changed the crossover function completely means where I was generating a random number from 0 to 19 there I changed it from 5 to 15 but it also could not work. In mutation, I changed each individual separately and implemented the given mutation function but still I could not get the linear graph.

6 Conclusion

In this report, I proposed an evolutionary algorithm dependent on the concepts and theory of natural selection and evolution by Charles Darwin. Evolutionary Algorithm has a similar idea which imitated the organic construction of the normal world dependent on Darwin's rules that are comprised of three tasks, i.e., selection, crossover and mutation. The evolutionary Algorithm is also based on natural selection theory as mentioned above. It is worth focusing that this complete evolutionary algorithm and its functioned have been programmed in MATPlOTLIB of Python programming language. This algorithm is fundamentally profitable because of its emphasis on the hunt space's better region, which came about because of a decent investigation double-dealing balance.

7 References

1. Musatafa Abbas Albadr 1,*, Sabrina Tiun 1 Masri Ayob 1 and Fahad AL-Dhief 2 1 CAIT, Faculty of Information Science and Technology, University Kebangsaan Malaysia

2. Osman and G. Laporte, "Metaheuristics: A Bibliography" Annals of Operations Research, Vol. 63, 1996, pp. 513–623.