Template Matching Problem

Table of Contents

Statement:
Given a larger and a smaller template image, you have to write a complete Evolutionary. Algorithm that takes the two images and solves the template matching problem, i.e. Find the coordinates or location of the smaller image within the larger image.

Scientific Model:
Scientific Models are generated or comes after deep learning about some natural Phenomena. If we wants to learn about what is scientific model or how can we make it then we need to go through from some processes to make such models, which are very useful in today world. Scientific approaches or processes are used to make such models or applications. If we learn about one simple model that how can it works? Then we see that they had huge backgrounds which make it scientific. In background working, we see that natural phenomena is exist which take huge time to complete one cycle (environment changes). On the basic of this, we make some theories which helpful in learning behavior. Then we analysis the behavior such theories how can we make the models which exactly shows such behavior as natural phenomena show. Then we make the applications which are based upon such models are helpful for human used.

In above statement, we have a problem which need such type of processes to make the solution. But in this case, we solve it in computer scientific way which need some computation or calculation. We need the Darvin theory which helpful to make computational methods for solving the problem.

Darwin Theory:
Darvin theory describe that how generations evolve over generation through inheritance trials. (Evolve means that how can generations changes over time, how can new generation comes from pre-existing species). According to the theory, one generation have some traits which enable them to adopt to their environments will help them to survive. If generation have less adaptive traits will less survive to pass them on. Passing time, these traits enable current

generation to survive and reproduce new generation which contribute more traits in the population and the population will evolve. We said that concept called **Natural Selection.**

For example:

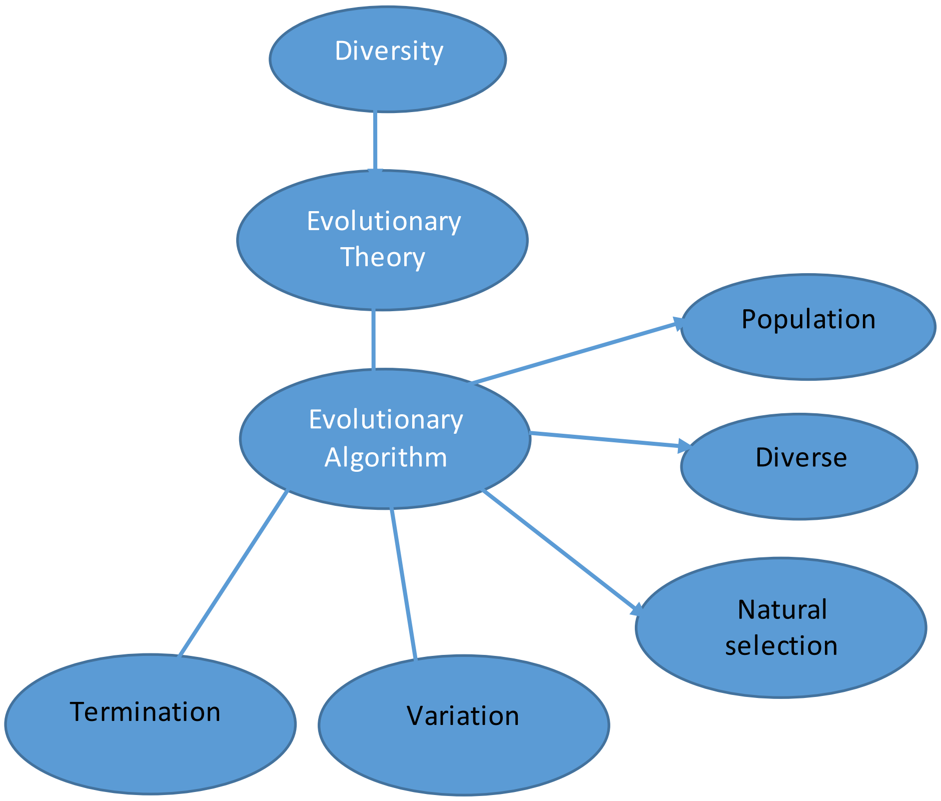
Darvin observed that the species of organisms on different place were clearly similar but had distinct difference which make them unique in their environment. Like seed-eating finches had stronger, thicker beaks for breaking seeds, while insect-eating finches had spear-like beaks for stabbing their prey. So, they are best suited to survive in their environment.

On the basic of these natural selection, generations survive in current environment and make some contribution to the next generation which help to survive on that environments. So, our above problem are based upon this theory. This theory contribute our model and application.

Computational models:

Some computation models which used these concept of natural phenomenon and make a reality based models, now, we will try to make such type of one simple model. In above theory, we see some diversity (some common traits like features of parent generation which makes us unique, able to survive in changing environment) which enables the generation an able to survive in environment over long period of time. So, we used diversity as our first component to make computation/ scientific model. On the basic of this diversity process, we make an **evolutionary theory** which saves the process of diversity for computation. We used the word “evolutionary” in our theory because diversity change over time. We need to make an evolutionary algorithm which based upon the theory.

In this evolutionary theory, we have a population which survive in current environment and evolve after passing certain period of time. On the basic of some trait (unique factors) which diverge them to the next generation. This unique factor make it a common to the whole population and help us to meet our problems requirements and make the solution.



Evolutionary Algorithm:

Natural Reality:
Adaptation of some natural phenomena (some process/ changes of environment over time without man interruption) on the basic of some computation, mathematical, scientific models, is called natural reality. There are some models which show natural reality. For example atoms are too small to observe directly, so models are used to visualize the part of atoms Rutherford, Bohr models etc. Evolutionary Model/algorithm which helps us to solve our template matching problems.

Template Matching Problem:
On the basic of evolutionary algorithm, we are trying to solve our above problem is just like a template matching problem. So, we need to make an algorithm according to the selection which is based upon some uniqueness which lies in their parent templates.

These are the some components which are used as the best suited to make an evolutionary algorithm.

Population generation

Correlation between the current population and parent population

Fitness level of generated population as compared to the original population

Generated new population which have some unique factor of their parent population.

Model of Template Matching:

According to scientific model, Model comes after theory because model can be concluded after having some theoretic data not just thoughts. Many theories exist that proved to be wrong and vice versa. For the Natural Reality phenomenon, Diversity in Nature is covered by a theory proposed by Charles Darwin in which he tried to explain how diversity comes in nature. Theory was explained above, it turns for the model. Theory explained a Phenomenon and Model will explain the theory. For this, Evolutionary Algorithm explains the theoretical data. As by name Evolutionary that means something will evolve in this model. Let's come to the spicy words that will be use in the model.

- Population – generation (first or current)
- Fitness – measured the survival
- selection – as nature do natural selection

These whole theory evolves nearer to these three words.

Attempts

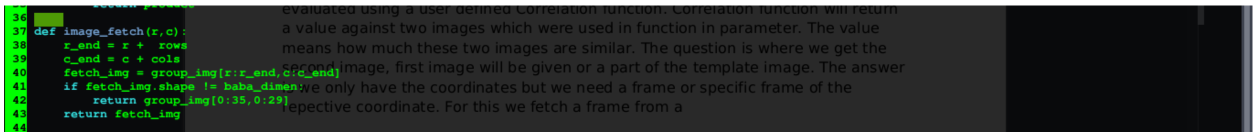
As per theory, there should be existence of some organism that can mutate itself to more or will generate more with another organism. For the algorithm, we need to produce the generation at first. But how will it look like?

Starting of the Evolution (First Generation)

A+s image is spread out with pixels on 2d frame. Every pixel will have a coordinate attribute. We all know that 2d frame can be measured on xy plane. So every pixel have the value of x and y to get located on the frame. For the generations, we attempt to provide the xy values in a tuple range will be the limited space of the image. For the x which will for the row/height will be limited to the 512 and y will be the column/width will be limited to the 1024. These values will be the generated randomly because we are unknown of the fitness level of the xy coordinate which we will choose. The population size will limit the generation size. The larger the population size or smaller size of population size will affect the processing time. To overcome this, we choose a moderate value like 100.

Fitness Evaluation

After introducing the generation on the plane, we must check the fitness level for each individual in population. The fitness will be between -1 and 1 because it will be evaluated using a user defined Correlation function. Correlation function will return a value against two images which were used in function in parameter. The value means how much these two images are similar. The question is where we get the second image, first image will be given or a part of the template image. The answer is we only have the coordinates but we need a frame or specific frame of the respective coordinate. For this we fetch a frame from the template given with the problem statement.



```
def image_fetch(r,c):
    r_end = r +  rows
    c_end = c + cols
    fetch_img = group_img[r:r_end,c:c_end]
    if fetch_img.shape != baba_dimen:
        return group_img[0:35,0:29]
    return fetch_img
```

evaluated using a user defined Correlation function. Correlation function will return a value against two images which were used in function in parameter. The value means how much these two images are similar. The question is where we get the second image, first image will be given or a part of the template image. The answer is we only have the coordinates but we need a frame or specific frame of the repective coordinate. For this we fetch a frame from a

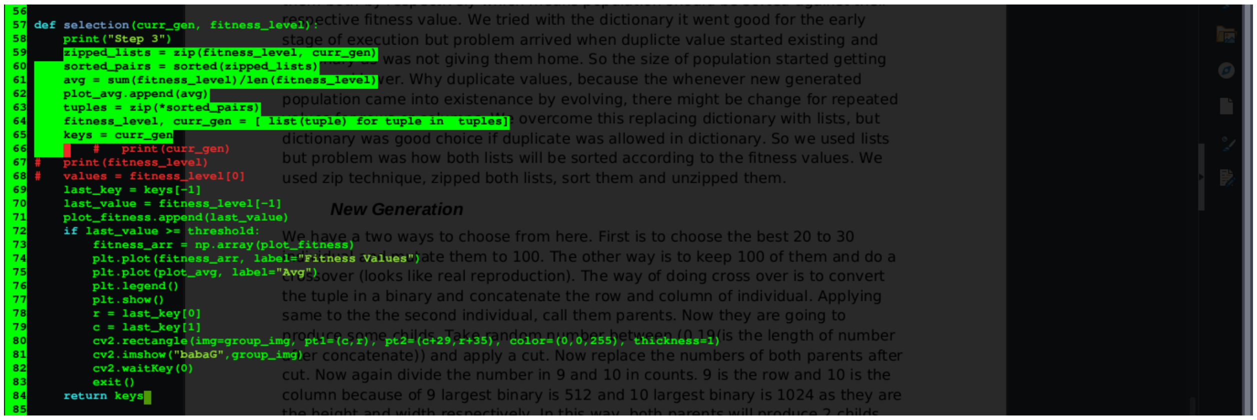
.

As code is written in the above screenshot. The image fetch function will get coordinates as in population and fetch the respective frame with same shape exactly as shape of the desired image. The function will then return a value that measures the fitness of respective individual. This function will be used for the 100 individual and then we will get the fitness value against each population individual. This fitness evaluation will return the list of fitness level.

After getting the fitness of the individual we need something that can select the fitter one as nature do natural selection given in theory.

Selection

Now everything we have is population and fitness against them. We need to sort them both by respectively which means population should be sorted against their respective fitness value. We tried with the dictionary it went good for the early stage of execution but problem arrived when duplicate value started existing and dictionary as was not giving them home. So the size of population started getting lower and lower. Why duplicate values, because the whenever new generated population came into existence by evolving, there might be change for repeated values for rows or columns. We overcome this replacing dictionary with lists, but dictionary was good choice if duplicate was allowed in dictionary. So we used lists but problem was how both lists will be sorted according to the fitness values. We used zip technique, zipped both lists, sort them and unzipped them.



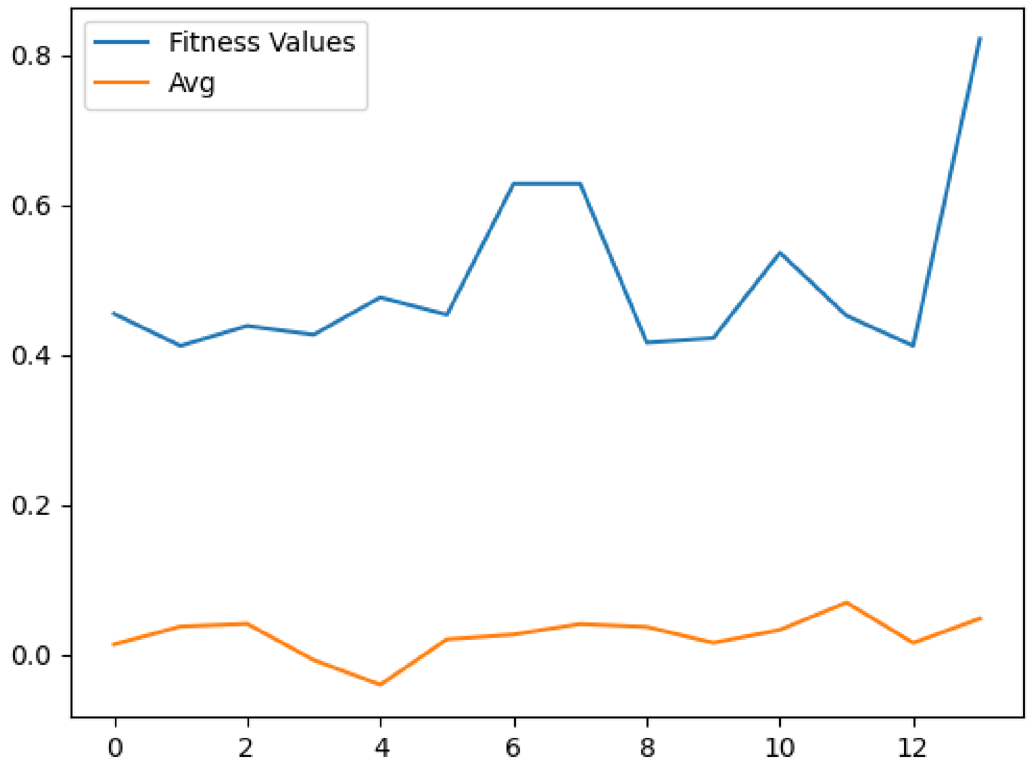
```python
def selection(curr_gen, fitness_level):
    print("Step 3")
    zipped_lists = zip(fitness_level, curr_gen)
    sorted_pairs = sorted(zipped_lists)
    avg = sum(fitness_level)/len(fitness_level)
    plot_avg.append(avg)
    tuples = zip(*sorted_pairs)
    fitness_level, curr_gen = [ list(tuple) for tuple in  tuples]
    keys = curr_gen
        #   print(curr_gen)
#   print(fitness_level)
#   values = fitness_level[0]
    last_key = keys[-1]
    last_value = fitness_level[-1]
    plot_fitness.append(last_value)
    if last_value >= threshold:
        fitness_arr = np.array(plot_fitness)
        plt.plot(fitness_arr, label="Fitness Values")
        plt.plot(plot_avg, label="Avg")
        plt.legend()
        plt.show()
        r = last_key[0]
        c = last_key[1]
        cv2.rectangle(img=group_img, pt1=(c,r), pt2=(c+29,r+35), color=(0,0,255), thickness=1)
        cv2.imshow("babaG",group_img)
        cv2.waitKey(0)
        exit()
    return keys
```

Here we placed a threshold, if the last value of the fitness level list, which is the highest value in all over list because it is sorted, is greater than the threshold value, just need to draw a rectangle having same shape as desired image on template with found message and then stop the program.

New Generation

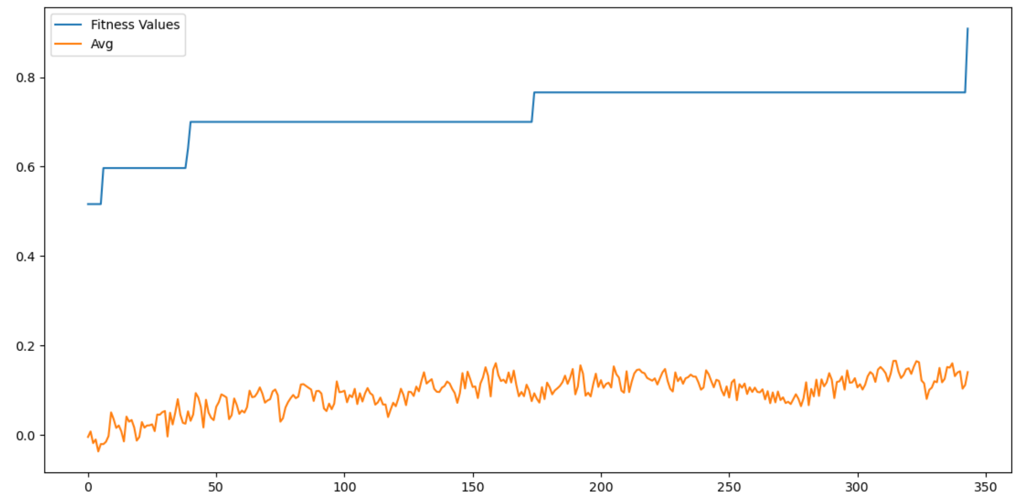
We have two ways to choose from here. First is to choose the best 20 to 30 individual and mutate them to 100. The other way is to keep 100 of them and do a crossover (looks like real reproduction). The way of doing cross over is to convert the tuple in a binary and concatenate the row and column of individual. Applying same to the second individual, call them parents. Now they are going to produce some child's. Take random number between (0, 19(is the length of number after concatenate)) and apply a cut. Now replace the numbers of both parents after cut. Now again divide the number in 9 and 10 in counts. 9 is the row and 10 is the column because of 9 largest binary is 512 and 10 largest binary is 1024 as they are the height and width respectively. In this way, both parents will produce 2 child's in my case but you can choose allow them produce multiple children, I wanted to keep the population size same as first generation but in nature there can be population size greater than or lower than the population size. This user defined function will produce 100 population size and then will return it.

After having the new generation we will start the loop till the fetched frames crosses the threshold, it can be other than the desired image. And then generations and generations. The graph of the of the my code is;



Above Figure shows a graph that shows fitness level and average value of the fitness level of the generation. This graph shows that fitness value varies respective to the generation. Sometime it got higher and sometimes it got lower, this happens because there may come a generation that have the maximum of fitness lower than the previous. It should evolve with time but it is not getting so.

It should display the graph as it is below, but that code is modified and applies some condition on the graph not on the actual evolution. We tried to make the code efficient but could not make it so, we tried to make the generations having the maximum fitness value that should evolve and get better and better with evolution.



Limitations

This code has some limitations which stop and limit this code from increasing its scope. The scope depends on the logics and code. This code is not much efficient and this results that sometimes it take much time in execution and time to. Reach the target. This can be done if code took the best 30 or 40 individuals and proceed with them and produce more from them and same procedure can be repeated again and again. Through this, there is possibility that results would be changed from now. Limitation of my code may include that if image is from on very side of the image, it may be unable detect it.

Application

Application can be fetched by viewing the model. One model can be used for variety of applications like sorting it can be used for various items, data, number etc. Similarly the evolutionary algorithm also have the large scope in efficient searching.

The scope of the code can be reached to the live image capture and finding a desired frame in it just like a satellite image that may have large amount of data in it. Evolutionary algorithm can help in finding it. It is not only searching a image, it can be something, after searched many things can be done. But the best was the template matching problem with our experience for now.