EC504 final project

Image Segmentation

Team: Gaomeizhu Qu / Wenqian Ma / Chenxi Zhang / Yihan Li

1. Introduction of the project

The project is going to address the problem of segmenting an image into regions. There are the following steps of our algorithm:

Input the image and calculate it into RGB-value, then turn the value into grey value. And set two grey values as the center points' feature, which called foreground and background.

Using K-means algorithm to cluster every pixel in the image into a stack which belongs to foreground or background. For each pixel i, we have likelihood values ai which belongs to foreground and bi which belongs to background. ai and bi are parameters for determining if the pixel belongs to foreground or background in isolation. If ai > bi, i belongs to foreground, and background otherwise.

Suppose V be the set of pixels in the image. We need to declare certain pairs of pixels to be neighbors which are contained in the set E. For each pair(i, j) of neighboring pixels, separation penalty pij >= 0 is used for determining the edge of object in the image.

The object determined by ai and bi and pij is the result of the algorithm.

2. Relevant reference

Algorithm Design by Jon Kleinberg, Eva Tardos (Chapter7.10 Image segmentation)

K-means definition:

http://stanford.edu/~cpiech/cs221/handouts/kmeans.html

How to find the center point of a cluster:

http://stats.stackexchange.com/questions/91065/finding-the-cluster-centers-in-kernel-k-means-clustering.

Video of K-means and K-means++:

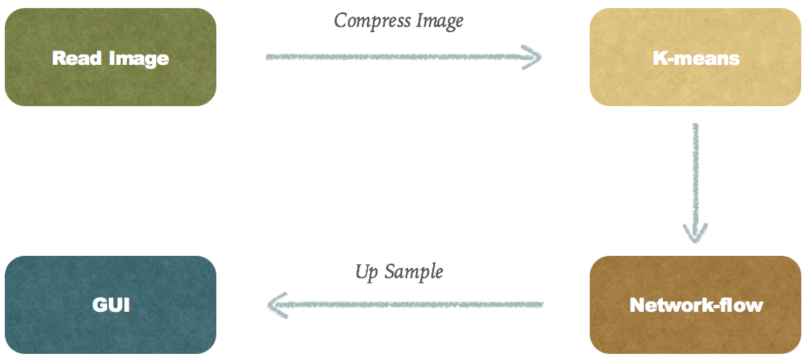
https://www.youtube.com/watch?v=yR7k19YBqiw&t=28s

Video of Ford-Fulkson algorithm:

https://www.youtube.com/watch?v=PY0LL_-jqa0

3. High-level design

The project’ s system diagram is as follow:



4. Project features:

a. Read an image as a matrix.

b. Compress original image into a new image with total 1000 pixels. The new image has the same length-width ratio.

c. Resize Image1 into a 1 dimension matrix named x, and goes into K-Means algorithm. The One output of it is a label matrix with only 1 and 0.

The other output of it are the values of two centers.

d. Calculate a_i and b_i.

a_i is the difference of the value of x[i] and center1

b_i is the difference of the value of x[i] and center2

e. The graph is a n*n matrix where n is equal to the number of pixels+2 (sink and source)

Penalty is defined by the difference of two pixel value.

Run Ford-Fulkson algorithm

The pixel whose a_i = 0 belong to set A

Others belong to set B

f. Enlarge the compressed image to the original size

g. Use Tkinter to build interface of the algorithm

5. Implement of project features

a. Read an image as a matrix.

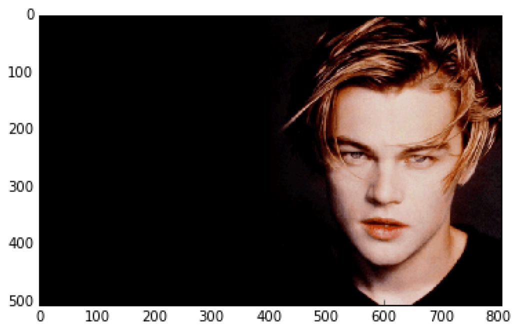
The first step of the project is to read an image in python to get ready for image segmentation. We used ‘imread’ function from ‘skimage’ package and read the image as two matrixes. The first matrix named ‘image’ which is a [col * row] matrix and the elements of the matrix is the grey value of every pixel. The second matrix named ‘imageRGB’ , which is sized [col * row *3] which is a matrix with RGB value of every pixel of the image.

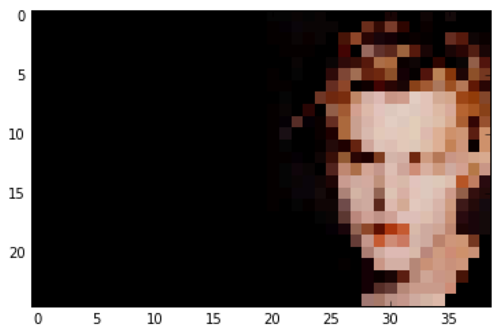
b. Compress original image

Because the running time of the whole program is depended on the size of the image, which may take several hours to several minutes, we decided to down sample the image into a fixed size before doing the segmentation part.

By recording the length-width ratio of original image, we compress the image into a fixed-size image which has about 1000 pixels. In this way, no matter how big the input image is, the running time of program stay almost constant.

The output of this part is two matrixes, one is a matrix with grey value and the other one is with RGB value.



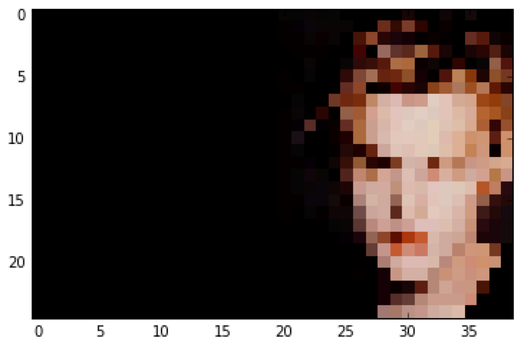


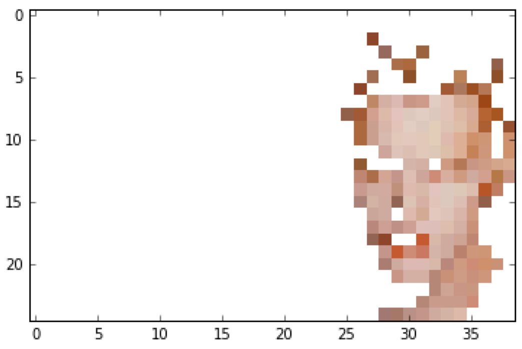
c. K-Means

We chose k-means method as the algorithm for clustering. K-means works as follows: it chooses k center points randomly at first, and compare the distance between each left point and the centers, and then set the point to the nearest center set. Every time we repeat this way, the centers will change , according to the new sets. We repeat this way over and over again, until there is no center changing.

In the image segmentation project , distance is the difference between each pixel and center.

In order to set all the pixels to either background or foreground, we need to divide all the pixels into two clusters according to their grey values. So we set k to be 2, and call k-means function. It returns a matrix with either 0 or 1 as elements, representing foreground or background. And it can also give the final center values, from which we can calculate ai and bi.





d. Calculate likelihood ai and bi

The output of K-Means is two centers value and a label matrix. After searching for information, we decided to use the Euclidean distance between pixels and centers to represent the likelihood a_i and b_i.

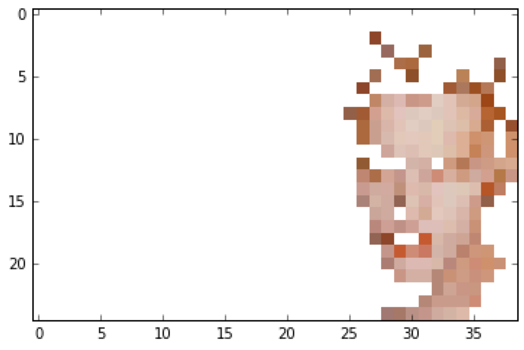
e. Network Flow

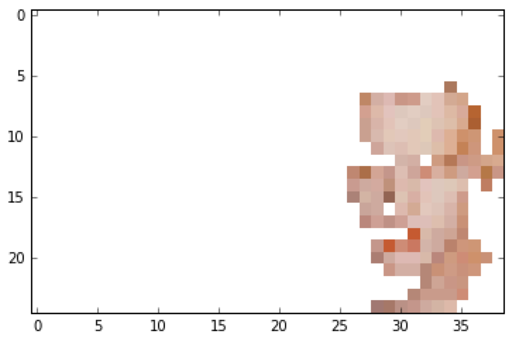
The input of the Network Flow is a matrix sized [n*n], which represents the graph we are going to run the algorithm, where n is equal to the number of pixels+2 (sink and source). For each pixel, the capacity between pixel and source is defined as a_i, the capacity between pixel and sink is defined as b_i.

We first considered Ford-Fulkson algorithm to run the Graph, and get the max flow of the graph. The penalty of this algorithm is defined by a function, which is defined after several times of trying. We first set the penalty as a fixed value, however it didn' t work very well. So we then considered defined penalty as a function which is set as the difference between current pixel' s a_i and the a_i next to this pixel. Due to the rules of Ford-Fulkson algorithm itself, after running the graph, there is a path that the flow is 0 exist in the residual graph. For the pixels whose likelihood a_i is 0, they belong to set A, which is considered as foreground or background of the image. On the other hand, the pixels whose a_i is not 0 belongs to set B. In this way, the image is separated into two different part: set A and set B.

Beside the Ford-Fulkson algorithm, we also considered about push relabel algorithm which has more smaller time complexity. Push-Relabel algorithm is an optimized one comparing to Ford-Fulkerson. Throughout its execution, the algorithm maintains a "preflow" and gradually converts it into a maximum flow by moving flow locally between neighboring nodes using push operations under the guidance of an admissible network maintained by relabel operations. Every node has another property—height. For each node, if there is a preflow value, and the capacity of associated edge in the residual graph is positive, then find a smallest height, set the height for this edge to be height + 1, and push min{preflow value, residual capacity} to that neighboring node.

Our push_relabel algorithm has a small bug,.Due to the time constraint, we applied ford-fulkerson eventually, but I 'm sure we can fix it after final exams.





f. Build application

Tkinter is a very handy library that we can use for developing GUI.

First, it reads an image, and store it as a matrix.

Second, when we click somewhere, "Event" can return the mouse coordinate (x, y) .

Third, according to the coordinate, it's easy for us to get whether we need to remove the background or foreground. We can just set the background or the foreground pixels in the processed matrix to be 0. And then use "config" function and "root main_loop()" to update the image shown in GUI.

6. Conclusion

The project may have some insufficient but we have try our best to finish it. The GUI we are using for presentation is not the final version. It only includes the K-Means part to show the method of using the application. The entire program of this project is attached in the folder we submit.

Also, although we lack the push relabel algorithm to make the project performs better, we still tried our best to optimize the efficiency of the project.