Name: PRATHMESH R. KUMBHARE

CS6001 Assignment 7: Graph Models

REPORT

A. OUTLINE FOR PROGRAM ALGORITHM:

The algorithm for implementing the program is given below in steps:

1. Images required for this assignment are taken from **assignment 1** of this course. The Matlab file **'Chain_test.m'** determines the class of each pixel in the image. (Face/Background) Training images as well as testing images with predicted pixel class are shown as visualization.
2. Factors which can be varied to get different results for a model are: values of Potts model.
3. The program firstly creates 'bwMask' for the training images. Then, the program loops through each row and column of 'bwMask' & increases the count of probability 'Pr_x_given_w_equalsTo_1' or 'Pr_x_given_w_equalsTo_0' if the value is 1 or 0 accordingly. Then these values are normalized. These values are **unary cost** for face nodes and background nodes correspondingly. Similarly, **pairwise cost** to the path between nodes is assigned according to Potts model.
4. For testing images, the program assigns unary cost to each nodes and also pairwise cost between them from the values calculated in the loop explained above. (Negative value is assigned to the unary cost values.) Another matrix named **Final_Cost_Matrix** created by adding both these costs at each nodes.
5. Then, a **directed graph** (digraph) is created using Matlab function. All the costs calculated in the Final_Cost_Matrix are assigned as edge weights to this network. The function **'shortestpathtree'** computes all the states (w=1/0) for each pixel in the network. It should be noted that the algorithm takes ones test image at a time and then each row of this image at one time.
6. The program creates 'detMask' by looping through each row and column of the mask and assigning value 1 or 0 depending on which class is predicted by the function. 'gtMask' (ground truth) is the annotated testing images. To calculate TP, FP & FN the program loop through rows and column of these masks and apply mathematical relations between them.
7. In the last part precision, recall & f_score is calculated from the above values of TP, FP & FN. Visualization for the predicted pixel classes is also provided.

B. RESULTS:

This is the table for all the models implemented using the program with all the important values:

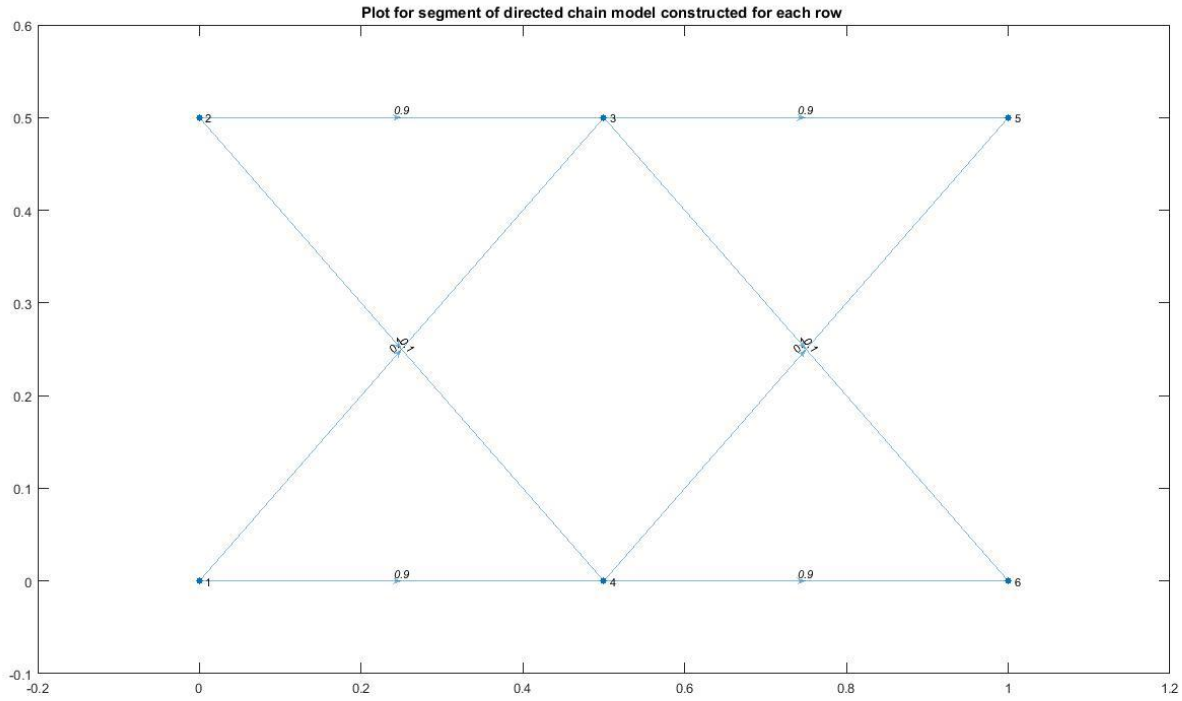
	MODEL	POTTS MODEL	PRECISION (%)	RECALL (%)	F_SCORE
1	RGB (Assignment 1)	-	66.48	64.45	65.45
2	RGB	[0.1,0.9;0.9,0.1]	23.78	30.98	26.91
3	**RGB**	**[0.5,0.5;0.5,0.5]**	**37.06**	**92.62**	**52.94**
4	RGB	[0.46,0.54;0.54,0.46] (Expected for this value)	4.48	49.47	8.22
5	RGB	[0.48,0.52;0.52,0.48]	32.80	60.95	42.65

Results for each model are given below separately.

1. **RGB Model:**

This model uses values of red, green and blue to describe the image.

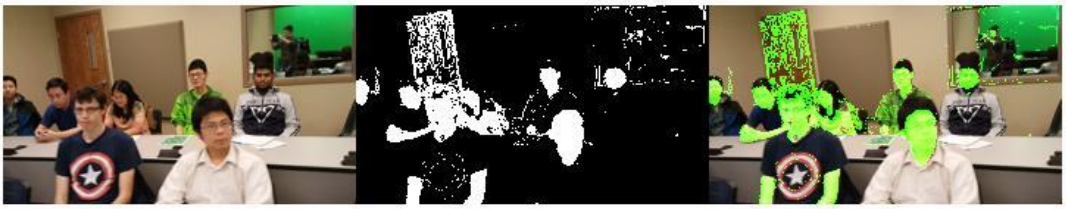
Network used for computing optimal path:



1. The upper row nodes represent the face class pixels and lower row nodes represent background class pixels.

Some Training Images:

Some Testing Images:



C. General Observation & Comments:

1. From images and table, we can easily see that this algorithm predicts the pixels with **satisfactory** accuracy compared to the model implemented in assignment 1.
2. As the values for Potts model are varied the results change drastically.

D. Summary and Concluding Comments:

1. The aim of this assignment was to classify pixel class in the given images. From the results, it is evident that the algorithm gives **marginally acceptable** results.
2. Different chain models can be used to construct the network. We are only applying this algorithm to each row of image. Larger areas, and hence complex connection networks can be considered to improve results.
3. Another important point to note is that according to instructions for assignment, **smoothing of image** is recommended to eliminate discrete boundaries of pixel class. I was unable to perform this operation for the algorithm. Inclusion of this function might improve the results significantly.
4. Perhaps, the most significant factor in this algorithm is the Potts model matrix. As this is the energy function, its values are calculated between each nodes. Since, I was unable to establish a definite relation for accommodating it in the program, it is taken as **constant.**