McGill University
ECSE 415 - Intro to Computer Vision
Assignment #3 - Due 11:59pm - November 10th, 2017

Preface

The purpose of this assignment is twofold: (1) to give you the opportunity to develop tools based on the Local Binary Pattern, (2) exploring various clustering methods for image segmentation. Please submit your assignment solutions electronically via the myCourse assignment dropbox. The solutions should be in **Jupyter Notebooks** format, see tutorial #2 for a reference. The complete assignment submission should be a zip file containing your Jupyter code (.ipynb) in addition to all output images. Attempt all parts of this assignment. The assignment is out of a total of **40 points**. The student is expected to write his/her own code. Assignments received up to 24 hours late will be penalized by 30%. Assignments received more than 24 hours late will not be marked.

Problem 1 - Celebrity Face Matching [20 points]

For this question you will be finding your celebrity look-alike. You will need a face image of yourself to use as an input image (see figure 3 for an example input image). I have attached a dataset of 20,000 celebrity images to which your picture will need to be compared. These images are taken from a small subset of the CelebA Dataset [2]. In terms of preprocessing, you should resize all images to (128x128) and convert them to grayscale. To compare face images, you will be implementing, by hand, a Local Binary Patterns (LBP) algorithm. The method we are using is based on the paper by T. Ahonen et al. [1]. The first step is to compute the LBP feature descriptor for your input image. The LBP operator assigns a label to every pixel of an image by thresholding the 3x3 neighbourhood of each pixel with the center pixel value and considering the result as a binary number. The histogram of the labels can then be used as a texture 256-dimension descriptor. Figure 1 below demonstrates this process. However, this process will only provide us with a global descriptor of the whole facial image. One improvement over this method is to use the texture descriptor to build several local descriptors of the face and combine them into a concatenated global descriptor. In order to do this, the input facial image is first divided into local regions and texture descriptors are extracted from each region independently. For each region, the local histogram is then normalized. The normalized descriptors are then concatenated to form a global descriptor of the face. Figure 2 shows examples of a facial image divided into local regions.

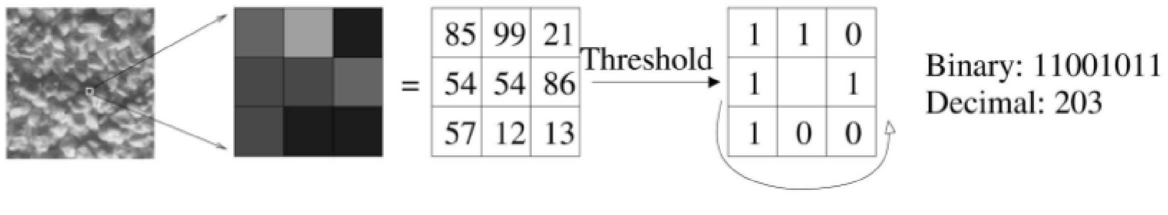

Figure 1 - The Basic LBP Operator [1]

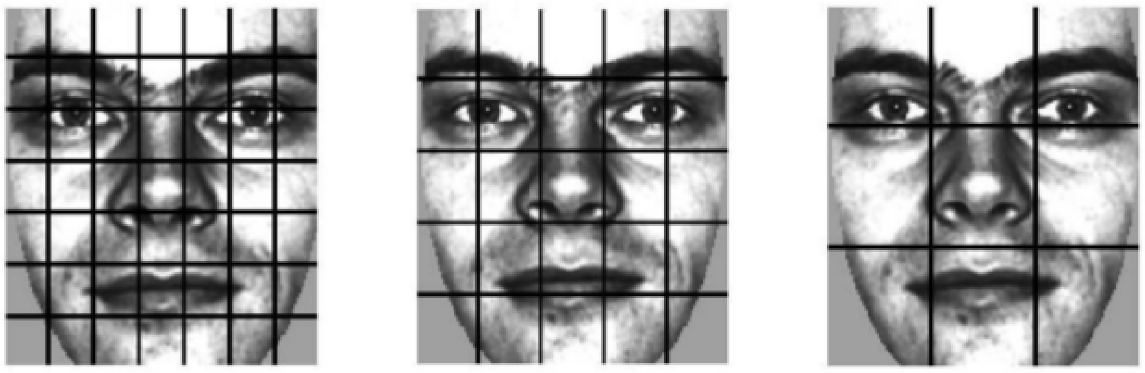

Figure 2 - Facial Image Divided into 7x7, 5x5, 3x3 Local Regions [1]

Once you have the descriptor of your facial image, you now need to cycle through all 20,000 celebrity images provided and compute their respective facial descriptors. You must then compare facial descriptors utilizing a sum-of-squared differences to find the closest celebrity image. Each image will have a dimension of 128×128, thus you will be subdividing your image into 7×7 cells. Each cell will have a dimension of 18×18 pixels ($width \times height$). The final facial descriptor will then have a dimensionality of $256 \times 7 \times 7$. The expected output should display your facial image along side of the best-matched image, as shown in figure 3. You may want to fine-tune these parameters depending on your input image, however, I found these parameters were fairly robust for various input images.

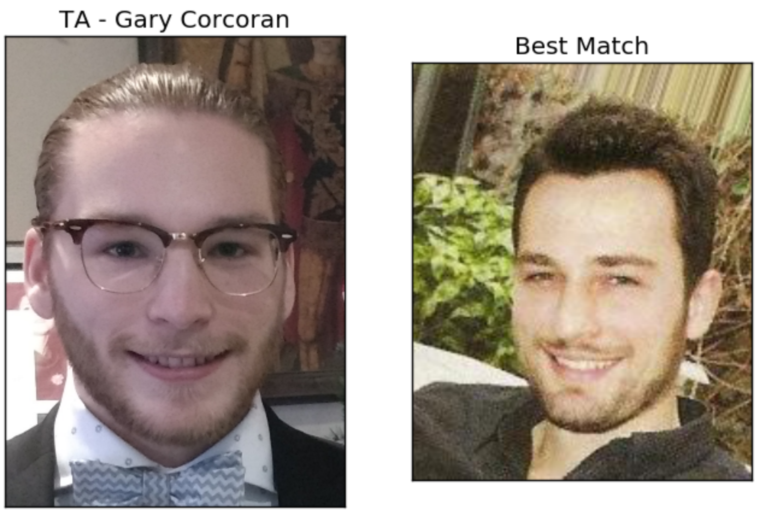

Figure 3 - Celebrity look-alike

Problem 2 - Image Segmentation [20 points]

For this question you will be performing image segmentation utilizing an image of your choice. You are required to implement, by hand, 2 different segmentation algorithms. The 2 segmentation algorithms you will be implementing are:

1. K-Means
2. Gaussian Mixture Models using Expectation-Maximization

For an overview of to K-means clustering algorithm refer to Lecture 11 - Slide 14. Additional, Gaussian Mixture Models using EM is explained in Lecture 12. For each of the 2 algorithms you will be exploring the intensity feature space (i.e. you should convert your image to grayscale using opencv's built-in functions). To build the Intensity feature space, you will be unraveling your input image into a vector of size $Width * Height$. The vector below demonstrates this process.

$$\begin{bmatrix} I_0 \\ I_1 \\ I_2 \\ \vdots \\ I_{W*H-1} \end{bmatrix}$$

For each of the two algorithms you will be segmenting your image into 2 and 4 clusters. Your end results should display the final segmented images. It is worth noting that depending on how you implement your algorithms you may experience long processing times, I would suggest downsizing your images to help counteract this. Lastly, based on your observations, which algorithm and number of clusters performed best on your input image?

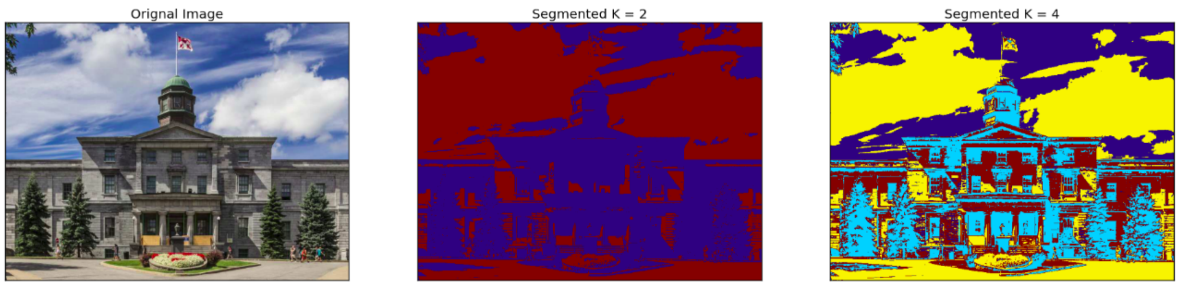

Figure 4 - Sample Image Segmentation using K-Means

References

[1] T. Ahonen, A. Hadid, and M. Pietikainen. "Face Description with Local Binary Patterns: Application to Face Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.12 (Dec. 2006), pp. 2037–2041. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.244.

[2] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*. 2015.