DS3002-Data Mining

Assignment #3

Spring 2024

Total: 100

Deadline: <u>30 March 2024</u>

Important Instructions

- ❖ You must complete the assignment before the deadline. Since the assignment is a little lengthy, get started right away!

- ❖ Plagiarism is strictly prohibited! Try to solve the assignment on your own. Marks will be given based on your thought process so make sure you have a solid reason for all your attempts.

- ❖ No assignment will be accepted after the due date.

- ❖ If Plagiarism Found, Straight ZERO in Whole Assignment.

Deliverables:

- Submission Format: i21-XXXX_A3.zip
- Submit 2 Jupyter Notebooks (Q1 and Q2 Separately) and Report

Marks Distribution

- Report 10 Marks
- Q1 (50 Marks)
- Q2 (5+20+5+10 = 40 Marks)

Q1. Interactive Foreground Segmentation Using K-Means Clustering

You will implement a basic version of the interactive image cut-out / segmentation approach called Lazy Snapping [1]. You are given several test images, along with corresponding auxiliary images depicting the foreground and background “seed” pixels marked with red and blue brush-strokes respectively. Your program should exploit these partial human annotations in order to compute a precise figure-ground segmentation.

You should first write a function that performs **K-Means Clustering** on color pixels. The input arguments are the desired number of clusters k and the data points to cluster. It outputs a cluster index for each input data point, as well as the k cluster centroids. You should then extract the seed pixels for each class (i.e., foreground and background) and use your k-means function to obtain N clusters for each class. A good choice for **N is 64**, but you can experiment with smaller or bigger values

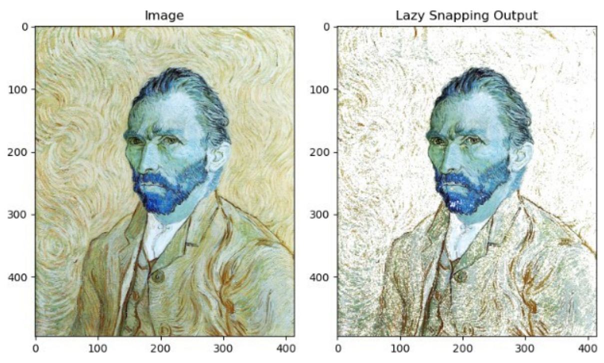
Next, compute the likelihood of a given pixel p to belong to each of the N clusters of a given class (either foreground or background) using an exponential function of the negative of the Euclidean distance between the respective cluster center C_k and the given pixel I_p in the RGB color space. The overall likelihood $p(p)$ of the pixel to belong to this class is a weighted sum of all these cluster. **For this Assignment w_k values will be 0.1**

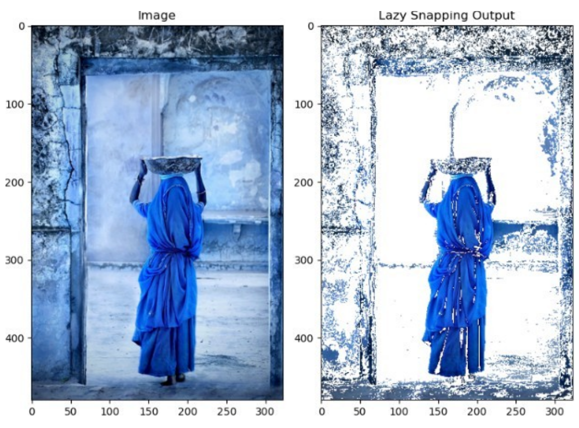
$$p(p) = \sum_k w_k e^{(-\|I_p - C_k\|^2)}$$

You might see improved results if you remove the square operation in equation above. The reason may have to do with the fact that squaring essentially reduces small values even further, mitigating (adversely) the effects of the negative exponential. Finally, a given pixel is simply assigned to the class to which it is more likely to belong. That is, if $p_{fg}(p) > p_{bg}(p)$, pixel p is assigned to the foreground class, i.e., $x_p = 1$, and vice versa.

Include your results for all test images in your report, and explain what you get. For test images with two stroke images, you should report results for both cases. Also compare results for different values of N, i.e., the number of clusters evolved in the foreground and background classes.

Results for N=64 (both foreground and background)





Q2. Face Recognition Using K-NN

For this question, we use simplified version of CMU Pose, Illumination, and Expression (PIE) Dataset, which only contains 10 subjects spanning five near-frontal poses, and there are 170 images for each individual. In addition, all the images have been resized to 32x32 pixels. The dataset is provided in the form of a csv file with 1700 rows and 1024 columns. Each row is an instance and each column a feature. The first 170 instances belong to the first subject, the next 170 to the second subject and so on. Following illustrate various instances of the first subject.

Tasks

1. Pre-process the dataset by normalizing each face image vector to unit length (i.e., dividing each vector by its magnitude). Next, for each of the 10 subjects, randomly select 150 images for training and use the remaining 20 for testing.

2. Implement a k Nearest Neighbors (k-NN) classifier from scratch and using the training set and evaluate its performance on the test set. You may not use built-in / library functions to implement the classifier.
    a. You should also implement Euclidean and cosine similarity distance measures and used them for different values of K.
    b. You should also present results when fewer training images are used (for instead 100 training images and 70 test images per subject)
    c. Write results in the report for k-values 2, 5, 7, and 11 with each distance metric.

3. Use Sklearn to apply SVM and GaussianNB on this dataset and compare the accuracy with K-NN in the report.

4. Perform dimensionality reduction on the training and testing datasets and visualize them in 3-D space using the Principal Component Analysis (PCA) and matplotlib or Seaborn.

Rubric

Criteria	Marks
Marks Distribution	
- Report	10
- Q1	50
- Q2	40
Q1. Interactive Foreground Segmentation Using K-Means	
- K-Means Clustering function implementation	15
- Seed pixel extraction and likelihood computation	20
- Experimentation and comparison with different N values	15
Q2. Face Recognition Using K-NN	
- Pre-processing and data splitting	5
- k-NN classifier implementation	20
- Evaluation with different K values	10
- Evaluation with fewer training images	5
- Comparison with SVM and GaussianNB	5
- Dimensionality reduction and visualization	10
Overall Presentation	
- Clarity and organization of the report	5
- Documentation and comments in code	5