ISYE 6740 Homework 2

1 Image compression using clustering [40 points]

In this programming assignment, you are going to apply clustering algorithms for image compression.

To ease your implementation, we provide a skeleton code containing image processing part. `homework2.m` is designed to read an RGB bitmap image file, then cluster pixels with the given number of clusters K. It shows converted image only using K colors, each of them with the representative color of centroid. To see what it looks like, you are encouraged to run `homework2('beach.bmp', 3)` or `homework2('football.bmp', 2)`, for example.

Your task is implementing the clustering parts with two algorithms: *K-means* and *K-medoids*. We learned and demonstrated K-means in class, so you may start from the sample code we distributed.

The file you need to edit is `mykmeans.m` and `mykmedoids.m`, provided with this homework. In the files, you can see it calls Matlab function `kmeans` initially. Comment this line out, and implement your own in the files. You would expect to see similar result with your implementation of K-means, instead of `kmeans` function in Matlab.

K-medoids

In class, we learned that the basic K-means works in Euclidean space for computing distance between data points as well as for updating centroids by arithmetic mean. Sometimes, however, the dataset may work better with other distance measures. It is sometimes even impossible to compute arithmetic mean if a feature is categorical, e.g, gender or nationality of a person. With K-medoids, you choose a representative data point for each cluster instead of computing their average.

Given N data points $\mathrm{x}^n (n = 1, ..., N)$, K-medoids clustering algorithm groups them into K clusters by minimizing the distortion function $J = \sum_{n=1}^{N} \sum_{k=1}^{K} r^{nk} D(\mathrm{x}^n, \mu^k)$, where $D(\mathrm{x}, \mathrm{y})$ is a distance measure between two vectors x and y in same size (in case of K-means, $D(x, y) = \|\mathrm{x} - \mathrm{y}\|^2$), μ^k is the center of k-th cluster; and $r^{nk} = 1$ if x^n belongs to the k-th cluster and $r^{nk} = 0$ otherwise. In this exercise, we will use the following iterative procedure:

- Initialize the cluster center μ^k, $k = 1, ..., K$.
- Iterate until convergence:
  - Update the cluster assignments for every data point x^n: $r^{nk} = 1$ if $k = \arg\min_j D(\mathrm{x}^n, \mu^j)$, and $r^{nk} = 0$ otherwise.
  - Update the center for each cluster k: choosing another representative if necessary.

There can be many options to implement the procedure; for example, you can try many distance measures in addition to Euclidean distance, and also you can be creative for deciding a better representative of each cluster. We will not restrict these choices in this assignment. You are encouraged to try many distance measures as well as way of choosing representatives.

Formatting instruction

Both `mykmeans.m` and `mykmedoids.m` take input and output format as follows. You should not alter this definition, otherwise your submission will print an error, which leads to zero credit.

Input

- `pixels`: the input image representation. Each row contains one data point (pixel). For image dataset, it contains 3 columns, each column corresponding to Red, Green, and Blue component. Each component has an integer value between 0 and 255.
- `K`: the number of desired clusters. Too high value of K may result in empty cluster error. Then, you need to reduce it.

Output

- `class`: cluster assignment of each data point in pixels. The assignment should be 1, 2, 3, etc. For $K = 5$, for example, each cell of class should be either 1, 2, 3, 4, or 5. The output should be a column vector with `size(pixels, 1)` elements.
- `centroid`: location of K centroids (or representatives) in your result. With images, each centroid corresponds to the representative color of each cluster. The output should be a matrix with K rows and 3 columns. The range of values should be [0, 255], possibly floating point numbers.

Hand-in

Both of your code and report will be evaluated. Upload codes with your implementation. In your report, answer to the following questions:

1. Within the K-medoids framework, you have several choices for detailed implementation. Explain how you designed and implemented details of your K-medoids algorithm, including (but not limited to) how you chose representatives of each cluster, what distance measures you tried and chose one, or when you stopped iteration.
2. Attach a picture of your own. We recommend size of 320×240 or smaller.
3. Run your K-medoids implementation with the picture you chose above, with several different K. (e.g, small values like 2 or 3, large values like 16 or 32) What did you observe with different K? How long does it take to converge for each K?
4. Run your K-medoids implementation with different initial centroids/representatives. Does it affect final result? Do you see same or different result for each trial with different initial assignments? (We usually randomize initial location of centroids in general. To answer this question, an intentional poor assignment may be useful.)
5. Repeat question 2 and 3 with K-means. Do you see significant difference between K-medoids and K-means, in terms of output quality, robustness, or running time?

Note

- You may see some error message about empty clusters even with Matlab implementation, when you use too large K. Your implementation should treat this exception as well. That is, do not terminate even if you have an empty cluster, but use smaller number of clusters in that case.
- We will grade using test pictures which are not provided. We recommend you to test your code with several different pictures so that you can detect some problems that might happen occasionally.

- If we detect copy from any other student's code or from the web, you will not be eligible for any credit for the entire homework, not just for the programming part. Also, directly calling Matlab function `kmeans` or other clustering functions is not allowed.

2 Spectral clustering [40 points]

1. Consider an undirected graph with non-negative edge weights w_{ij} and graph Laplacian L. Suppose there are m connected components $A_1, A_2, \ldots, A_m$ in the graph. Show that there are m eigenvectors of L corresponding to eigenvalue zero, and the indicator vectors of these components $I_{A_1}, \ldots, I_{A_m}$ span the zero eigenspace.

2. Real data: political blogs dataset. We will study a political blogs dataset first compiled for the paper Lada A. Adamic and Natalie Glance, "The political blogosphere and the 2004 US Election", in Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem (2005). The dataset nodes.txt contains a graph with $n = 1490$ vertices ("nodes") corresponding to political blogs. Each vertex has a 0-1 label (in the 3rd column) corresponding to the political orientation of that blog. We will consider this as the true label and try to reconstruct the true label from the graph using the spectral clustering on the graph. The dataset edges.txt contains edges between the vertices.

   Here we assume the number of clusters to be estimated is $k = 2$. Using spectral clustering to find the 2 clusters. Compare the clustering results with the true labels. What is the false classification rate (the percentage of nodes that are classified incorrectly).

3 PCA: Food consumption in European area [20 points]

The data food-consumption.csv contains 16 countries in the European area and their consumption for 20 food items, such as tea, jam, coffee, yoghurt, and others. There are some missing data entries: you may remove the rows "Sweden", "Finland", and "Spain".

Implement PCA by writing your own code.

1. Find the first two principal directions, and plot them.

2. Compute the reduced representation of the data point (which are sometimes called the principal components of the data). Draw a scatter plot of two-dimensional reduced representation for each country. Do you observe some pattern?