Assignment 3. k-Means Clustering for Image Segmentation and Compression

Due Date: **November 1, 11:59 pm**
Assessment: 5% of the total course mark.

Description:

In this assignment you are required to implement the k-means clustering algorithm and apply it to image segmentation and compression. This experiment is similar to the experiment reported in section 15.1.1 of [1].

Find an appropriate function in the matplotlib library to read a colour image as a 3D array. In this format, each pixel is represented as a point in the three-dimensional space, whose components are the intensities of red, blue and green channels (the colour of the pixel is determined by these intensities). The goal of image segmentation is to partition the image into regions each of which has a reasonably homogeneous appearance or which corresponds to objects or parts of objects [2]. You will do this using the k-means clustering algorithm. Your training set will consists of all the 3D-pixel values. (Each pixel in the image is treated as a separate data point in the training set, thus even if two pixels at different locations have the same 3D-value, they are considered different data points.) The clustering groups 3D-pixel values that have similar colours. After obtaining the clustering you have to reconstruct the image by using the center of the cluster as the reconstruction of a pixel. This way compression is achieved since the number if bits needed to store the image is much lower, i.e. only $\log_2 k$ bits per pixel instead of 8 or 16 bits per pixel for the original image. Note however, that more efficient compression algorithms for images are used in practice (e.g., JPEG).

You will have to write a python program and a report. The program has to be modular and contain instructive comments. Use vectorization where possible. In the report you have to present the results for two colour images of your choice. For each image, you have to run the clustering algorithm for $k = 2, 3, 10, 20, 40$ (at least) until convergence at least three times, each time with a different initialization. Use two initialization strategies: 1) randomly pick the centers from the data points (use this two times); 2) pick the centers such that they have a sufficiently large distance between them. You decide how to implement the second strategy and then describe it in your report.

In the report you have to include the original images and their reconstructions for the five values of $k = 2, 3, 10, 20, 40$ (at least) obtained whith each initialization of the algorithm. You have to also include the **root mean squared error (RMSE)** between the original image and its reconstruction for each case. Organize this information nicely in one or two tables. Additionally, include the number of iterations until convergence for each case (nicely organized in one or two tables).

Stopping criteria: Running the algorithm until convergence means choosing a small threshold θ and stopping when the relative chnage in the cost is smaller than the threshold, i.e., when

$$\frac{RMSE_{old} - RMSE_{new}}{RMSE_{old}} < \theta.$$

You may choose $\theta = 0.001$ or smaller. Indicate the value of θ in your report. If the algorithm takes too long to run, you may choose a bigger threshold or limit the number of iterations. Document this process in your report.

The report should contain a discussion of the results: for each image and each k, which initialization strategy led to better clustering judging based on a) the RMSE; b) the visual reconstruction? Does always a smaller RMSE correspond to a more pleasing visual reconstruction? Is one initialization strategy better than the other all the time or almost all the time? Include any other observations you might find useful.

Submission Instructions:

- Submit the report in pdf format, the python file (with extension ".py") containing your code, and a short demo video. The video should be 1 min or less. In the video, you should scroll down your code and explain what each part does. Submit the files in the Assignments Box on Avenue.
- Naming convention: The names of your files should include: assignment number, student name and student ID number, for instance: "A3_Thomas_Edison_32415.py"

References

[1] C. M. Bishop, Deep Learning. Foundations and Concepts, Springer, 2024 (ISBN: 9783031454677), available for free at https://www.bishopbook.com/.

[2] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, Prentice Hall, 2003.