BOĞAZİÇİ UNIVERSITY

CMPE 537

Homework 1

Color Quantization Using K-Means Algorithm

Berke Esmer
2015400021

October 13th, 2019

1. Introduction

This assignment is given as the first project of CmpE 537 Computer Vision course for the term ‘Fall 2019.

The aim of the project is to get familiar with tools and libraries which are going to be used throughout the semester and apply K-Means clustering algorithm to images to obtain color quantization.

Color quantization is simply the process of reducing numbers of colors used in an image while trying to maintain the visual appearance of the original image.

2. Script Functions

There are three sections in the source code of project and all of them are explained below.

1. main function first sets the arguments used in terminal run and parses the input. Then it obtains the image from the given path and reads it. It differentiates the point selection mode by the input of terminal, and it proceeds with the algorithm by calling quantize function.

2. euclidean_distance function takes the image data and an RGB color and calculates Euclidean distance of all pixels to that color. At the end, it returns the distance array.

3. quantize function takes the raw image with the number of color centroids and iterations. It first extracts the attributes of image, then places it into a NumPy array reshaping it by (width * height, 3). Afterwards, for every pixel, first it calculates the Euclidean distance to every color centroid, secondly it selects the minimum distance and assign the pixel to its color centroid, finally it finds the new color centers by taking the mean of every cluster. This operation takes a loop of number of iterations and at the end, it either prints the image on screen, or saves it into the output folder.

3. How to Run

The script takes three required and one optional argument which all are explained below.

```
1.'-f', '--file', required=True, description="Input image path"
2.'-m', '--mode', required=True, description="1 is manual select, 2 is uniform random".
3.'-s', '--size', required=True, description="How many of k-means centroid"
4.'-i', '--iteration', required=False, help="Number of iterations, default is 10"
```

Mode is for the two deliverables of homework. ‘Mode 1’ runs by the manual selection of points, and ‘Mode 2’ runs by the NumPy Uniform Random selection.

```
e.g. python main.py -f ./data/1.jpg -m 2 -s 32
```

4. Output Images

Image outputs will be in the order of color quantization levels of *2, 4, 8, 16 and 32* with the *mode 1 and 2*.

Figure 1. Frog by using manual pixel selection [1920x1200]

Figure 2. Frog by using uniform random pixel selection [1920x1200]

Figure 3. Cat by using manual pixel selection [500x303]

Figure 4. Cat by using uniform random pixel selection [500x303]

Figure 5. Motorcycle by using manual pixel selection [1400x973]

Figure 6. Motorcycle by using uniform random pixel selection [1400x973]

5. Outcome & Conclusion

As far as concerning results, color quantization is critical for displaying images with many colors on devices that can only display limited number of colors, usually due to memory limitation. Especially, in the past, there were computers that had only a few kilobytes of memory and they had been still capable of displaying images. This operation also enables efficient compression of certain types of images. For instance, the first picture is 375 KB in its original form. Using color quantization level 2, the image size is reduced down to 196 KB. This is an outstanding result concerning the storage size. Of course, level 2 is not very appealing in terms of differentiating the details, but it shows how much we can compress an image by just using this operation.

Finally, there are just a few differences between manual and random selections. This difference would be reduced if we increase the iterations since their limit goes to the same average RGB value.