Digital Image Processing: 5th assignment

Tiago Moreira Trocoli da Cunha
RA: 226078
t226078@dac.unicamp.br

Jun 2019

1 Introduction

The objective of the fifth assignment is to reduce the number of colors of images via any clustering algorithm preserving the quality of them at the same time. In order to do that, the program has to load png images, apply a clustering method to find the most representative group of colors, save a codebook generated by the algorithm and reconstruct images using codebooks.

2 Program Specification

The code is written using Python 3+ in Ubuntu 18.04. I used the following libraries:

- **skimage**: used for load and save images.
- **sklearn**: used for applying k-means algorithm.
- **numpy**: used for matrix operation and manipulation.

The program takes images baboon.png, monalisa.png, peppers.png and watch.png as input, shown in figure 1. The program outputs three kinds of files. Codebooks are files containing clusters, their name format is codebook<name of image><number of cluster>.npy. Files with the word compressed" have the labels, their name format is compressed_<name of image><number of cluster>.png. And files containing images with 16, 32, 64 and 128 colors with name format as <name of image><number of cluster>.png.

Figure 1: From left to right: baboon, monalisa, peppers and watch images.

3 Code and Literature Review

The program uses k-means, a popular unsupervised machine learning algorithm that was initially developed for signal processing. It divides a set of n data $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$, $\mathbf{x}_i \in R^d$, into k clusters, $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_k\}$ in which each data belongs to the cluster with the nearest mean. In mathematical therms:

$$\underset{\mathbf{S}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in \mathbf{S}} ||\mathbf{x} - \mu_i||^2,$$

in which μ_i is the mean of points in $\mathbf{S}_i$. K-means algorithm iteratively refines the set of clusters and assigns each data to the nearest cluster. Given a set of k means $m_1^1, m_2^1, ..., m_k^1$ at first iteration, below is an overview of the algorithm:

Assignment step: assign each data to the cluster whose mean has the least squared Euclidean distance.

$$\mathbf{S}_i^t = \{\mathbf{x}_p : ||\mathbf{x}_p - m_i^t||^2 \leq ||\mathbf{x}_p - m_j^t||^2 \; \forall j, 1 \leq j \leq k\},$$

where each x_p is assigned to exactly one $\mathbf{S}_i^t$, even if it could be assigned to two or more of them.

Update step: Calculate the new means of the data in the new clusters.

$$m_i^{t+1} = \frac{1}{|S_i^t|} \sum_{\mathbf{x}_j \in \mathbf{S}_i^t} \mathbf{x}_j.$$

The algorithm stops when the assignments no longer change. Figure 2 depicts k-means steps.

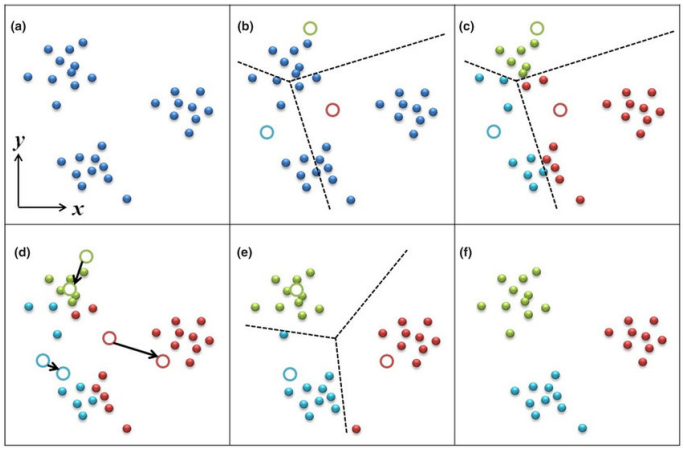

Figure 2: a) Initially all data belongs no cluster. b) Cluster are randomly created. c) Each data is assigned to the nearest cluster. d) Clusters are updated according to their data. e) Again, data are assigned to the (updated) clusters. f) K-means stops when no data is re-assigned to a new cluster.

4 Results

Seeing the baboon image with 16 colors only, we clearly can see a quality loss. Its left eye is red, which is not true, from the true image it is dark yellow. Also, its nose is unrealistically red (very bright red). With 32 color, the image becomes way better, fixing its eye but not its nose. I cannot distinguish any difference with 64 and 128 colors.

Figure 3: From top to bottom, left to right: baboon image with 16, 32, 64 and 128 colors.

Seeing monalisa with 16 colors, we clearly can see quality loss especially shadow regions. Our eyes can see levels of shadows, for example, is her neck. However, I cannot discern any difference with 32 colors and beyond.

Figure 4: From top to bottom, left to right: monalisa image with 16, 32, 64 and 128 colors.

Seeing peppers with 16 and 32 colors, we can discern differences. Colors are not the same from the real image and also can see a discrete variation of colors, in the real image these variations are more smooth. I cannot see any difference with 64 and 128 colors.

Figure 5: From top to bottom, left to right: peppers image with 16, 32, 64 and 128 colors.

Watch images suffer the major quality loss between all four images. With 16 color, the green and blue arrows turned to gray, the red arrow also changed. We can also see a quality loss in shadow regions. With 32 colors, we can still see quality loss, but at least the arrows are with a similar color from the true image. I cannot distinguish any differences images with 64 and 128 colors.

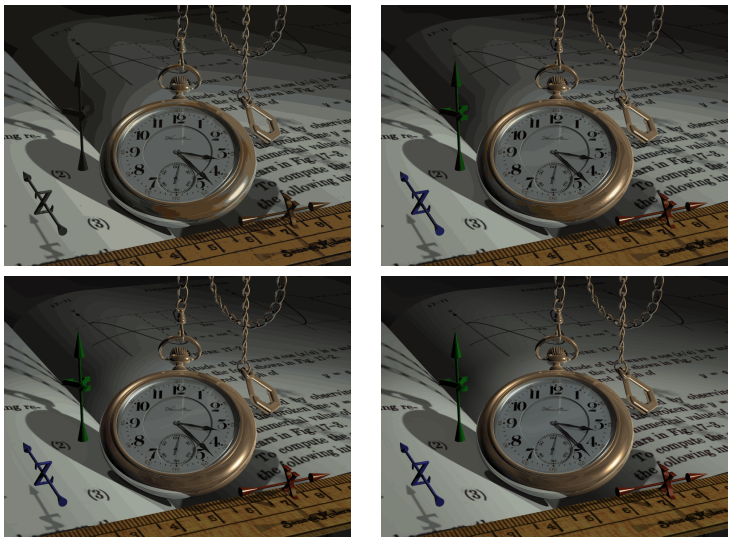

Figure 6: From top to bottom, left to right: watch image with 16, 32, 64 and 128 colors.