CS6375 Project 4 Part-1 Report
sxj220028

• Display the images after data compression using K-means clustering for different values of K (2, 5, 10, 15, 20).
- The output images after running K-means clustering on Koala.jpg and Penguin.jpg are present in the tests directory. The images are run with *k* in the set {2, 5, 10, 15, 20, 50, 100} and with varying *p* in the set {1, 2, 3}; the output images are named <Koala/Penguin>-<k>-<p>.jpg. Here *k* is the usual cluster count in K-means clustering and *p* is the order of Minkowski distance used to calculate the distance between 2 pixels. As usual, *p=1* and *p=2* correspond to Manhattan and Euclidean distance in the 3-dimensional space of Red, Green, and Blue values in the set [0, 255]. I wasn't sure whether to use Manhattan or Euclidean distance to calculate the distance between 2 pixels, thus I tried both and an extra *p=3* because there are 3 dimensions to color values.
To calculate the distance between 2 pixels where each is represented by a 32-bit integer value, I first extract the individual RGB components into their respective 8-bit values (using the *Rgb.getRgbIntVal function*) and then use the Minkowski distance of the order p to calculate distance.
The K-means clustering implementation works as per the class slides, I first select *k* random means from the RGB values in the *rgb array*. Then I assign each pixel in the *rgb array* to the respective cluster (*clusters hashmap*) based on which cluster center is currently the closest (using the *getClosestClusterCenter function*). Furthermore, after all the pixels in the rgb array are assigned, I update the cluster center with the average value of the pixels in the cluster (using the *calculateAverageRgbVal function*).

• What are the compression ratios for different values of K? Note that you have to repeat the experiment multiple times with different initializations and report the average as well as variance in the compression ratio.
-

K-value	**Koala [Image size (KiB): Avg, Variance]**	**Penguin [Image size (KiB): Avg, Variance]**
2	94.64, 87.248619	55.81, 366.86
5	171.2226, 9.3107843	92.01, 110.781268
10	165.29, 7.23	119.275, 75.151936
15	162.4553, 1.8578323	117.97233, 18.3823563
20	157.625, 5.3498529	119.034, 7.459183
50	151.6396, 0.0134203	115.97, 1.317028
100	149.44833, 0.144577	113.98966, 0.19062433

I have the different files which have the same k but different p as different runs and calculated their average and variance.

Original Image size of Koala.jpg = 762.5KiB
Original Image size of Penguin.jpg = 759.6KiB
Average Compression ratio for Koala.jpg (K = {2, 5, 10, 15, 20, 50, 100}): 8.06, 4.45, 4.61, 4.69, 4.83, 5.02, 5.10
Average Compression ratio for Penguin.jpg (K = {2, 5, 10, 15, 20, 50, 100}): 13.61, 8.25, 6.37, 6.44, 6.38, 6.54, 6.66

• Is there a tradeoff between image quality and degree of compression. What would be a good value of K for each of the two images?
- Yes, there seems to be a slight tradeoff but the relation is very unclear, as I've noticed that with increasing k value, it is not necessary for the resulting image will increase significantly in size.
But we can observe that as k increases the quality of the image also increases. Also, the quality seems to stagnate after a high value of K, as we don't see a huge difference in image quality of K=20, 50, 100.
So, overall K=50 or 100 seems to be a good value with respect to image quality and compression.

Note: I observed a slight difference in what colors are highlighted based on the p that we use in the distance calculation. I'm not sure what could be the reason for this, but I think it is an interesting thing to point out. This color difference due to varying p values is more apparent in Penguin.jpg.