MTH 552 : Statistical and AI Techniques in Data Mining

Submitted to : Dr. Amit Mitra

Image compression using Enhanced K-means Clustering

Submitted by:

Amit Badoni

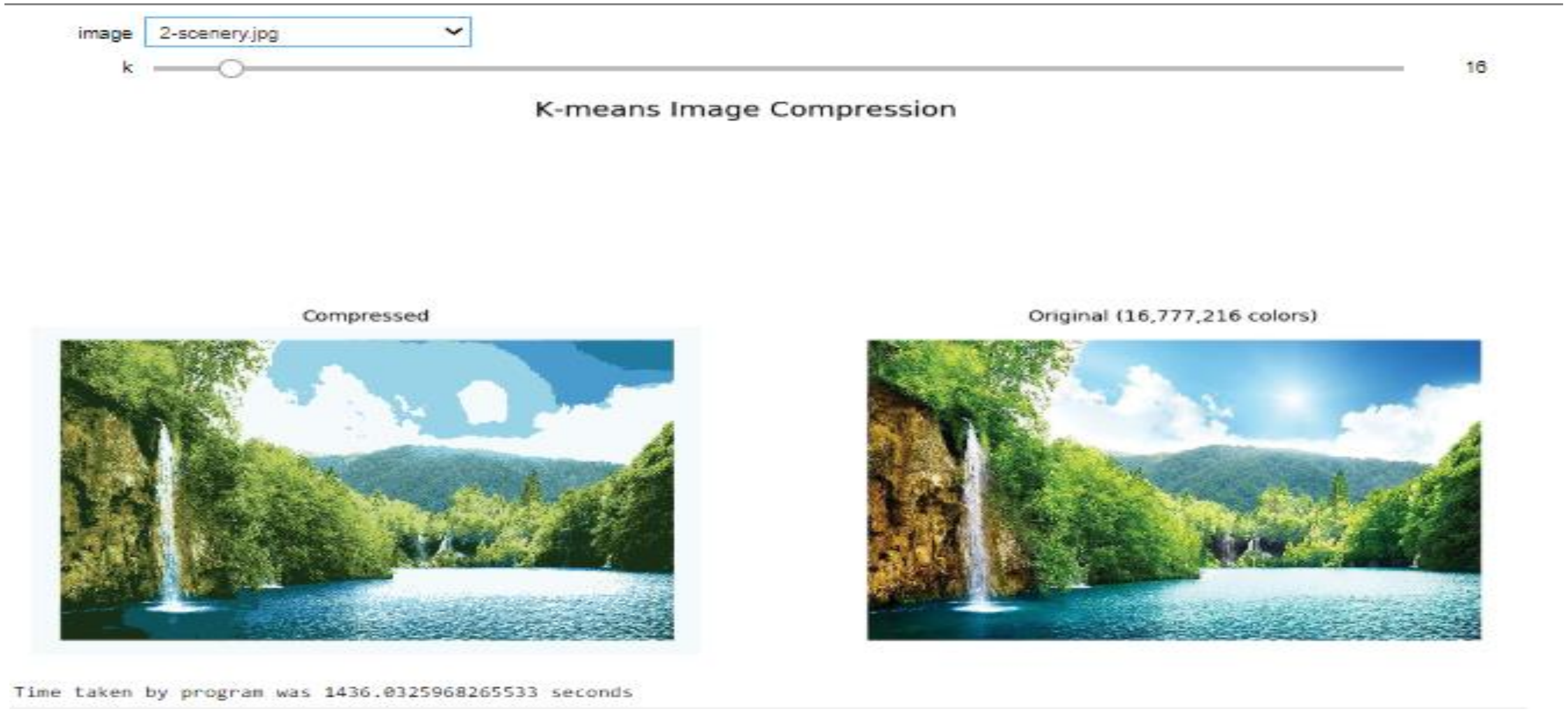
B.Tech Chemical Engineering

170091

Objectives:

1. Applying standard K-means algorithm to reduce image size
2. Enhancing algorithm for efficient computation with improved time complexity

Program Developed



```
Time taken by program was 1436.0325968265533 seconds
```

Work-Flow

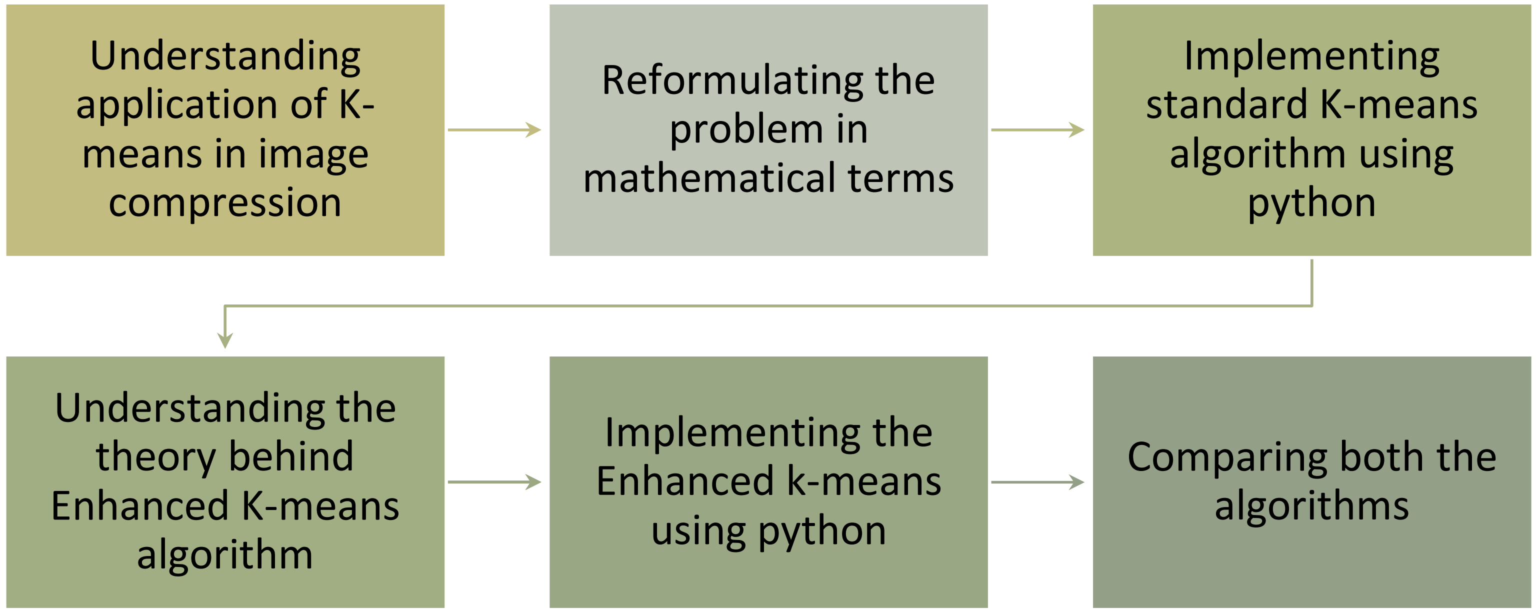

Understanding application of K-means in image compression
Reformulating the problem in mathematical terms
Implementing standard K-means algorithm using python
Understanding the theory behind Enhanced K-means algorithm
Implementing the Enhanced k-means using python
Comparing both the algorithms

Image Compression

- Images are stored on computer as a collection of tiny pixels.
- Width and height of image measured in pixels; Total number of pixels = width × height.
- Each pixel has a vector with 3 no.s from 0 to 255 which tells the computer the pixel's color
- The 3 no.s correspond to the intensity of Red, Green and Blue color respectively. The combination of the 3 colors and intensities give the pixel its color. This is known as RGB representation.
- K-Means algorithm forms a cluster of pixels. Each pixel in the original image is then replaced by the centroid of the cluster it belongs to leading to image compression.

Mathematical Problem Statement

"Given a set of $M \times N$ vectors, each of length 3, containing integers ranging from 0 to 255, perform k means clustering and replace the vectors with the centroid of their respective cluster"

Vectors correspond to the pixels of the image	**M = Width of the image**	**N = Height of the image**

Glossary

- Total number of pixels = T
- No. of clusters = k
- Database containing pixels = img_data
- Database containing centroids = Centroids
- Database containing cluster number of each pixel = Assignment vector

Applying Standard K-means

Table 1: Functions defined in the Program

S. No	Function Name	Input	Output
1	dist	2 Vectors(pixels)	Euclidean Distance between the vectors(pixels)
2	init_centroid	k	k random vectors which go in centroids
3	assign	Centroids, k, img_data	Assignment vector(size=T)
4	update	Assignment vector, k, img_data	Centroids(size=k)
5	compression	Image, number of clusters k	Compressed image and the runtime of function

Functions

- **dist:** Calculates Euclidean distance between 2 given vectors(pixels).
- **init_centroid**: Generates k random vectors that are taken as centroids to initialize the k-means algorithm.
- **assign**:

  1. Calculates the distance of every vector from every centroid and assigns each vector a cluster whose centroid is the closest.
  2. Returns the Assignment vector whose size is T.

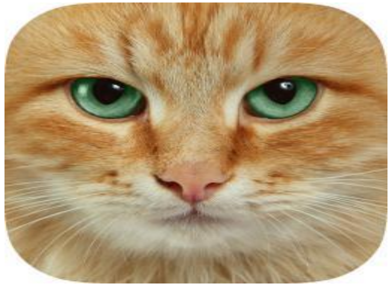
- **update**: Generates new Centroids by using img_data and Assignment vector
- **compression**:

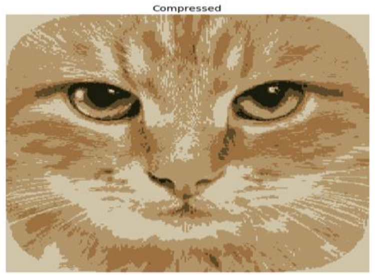
  1. Takes in image from computer and converts it into img_data.
  2. Normalises img_data
  3. Applies init_centroid to initialize the algorithm.
  4. Performs 5 iterations with each iteration having 1 assign and 1 update operation. *
  5. Replaces original pixels by their cluster's centroid using Assignment vector.
  6. Displays initial and final images along with the run time of the function.

* The standard exit condition (minimizing the sum of variances within cluster) requires lot of computations, thus could not be incorporated.

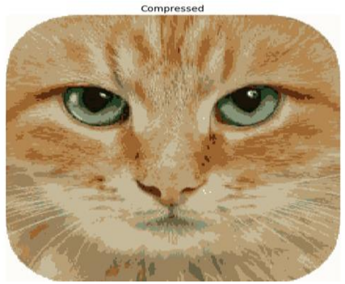
Results

- **Standard K means algorithm has high computation time even for 5 iterations.**
- **Not much quality improvement of 2nd image from 20 to 80 clusters suggesting 20 is either equal or close to the elbow of the elbow curve.**

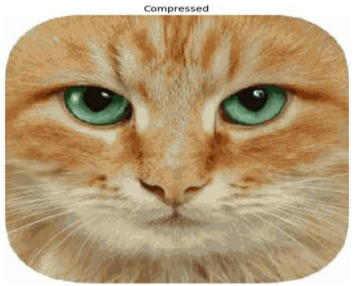

Original image
Dimensions = 255×255
Pixels = 65025



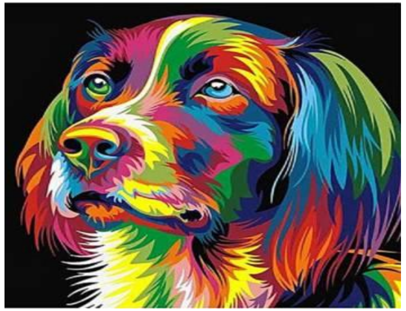
Clusters = 5
Time taken = 17.9 seconds

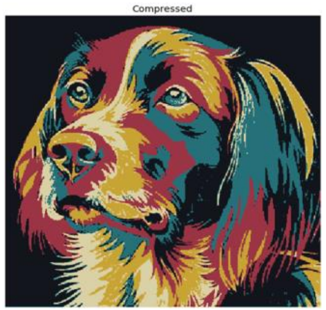


Clusters = 20
Time taken = 51.9 seconds

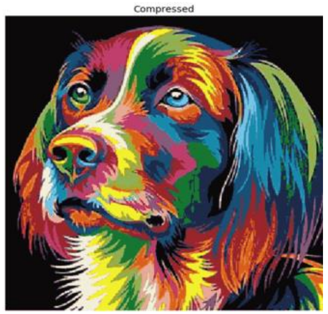


Clusters = 80
Time taken = 196.4 seconds

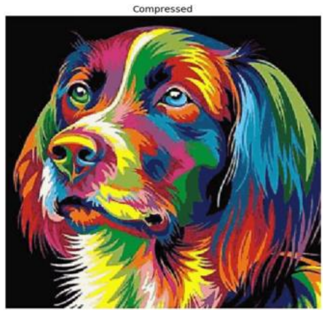

Original image
Dimensions = 352×352
Pixels = 123904



Clusters = 5
Time taken = 35.4 seconds



Clusters = 20
Time taken = 106.9 seconds



Clusters = 80
Time taken = 406.3 seconds

Enhanced K-means

The current algorithm can be enhanced by 2 ways:

1. Improving Accuracy:
   - Performance depends upon the centroids chosen initially.
   - Clusters can be different based on initial centroids leading to a local rather than global optimum.
   - Carefully chosen centroids can give better results.

2. Efficiency:
   - Each iteration consists of 2 steps:
     1. Assigning each point a cluster.
     2. Calculating the new centroid of each cluster.
   - Number of computations are fixed in step 2, but can be reduced in step 1 to improve time efficiency.

Improving Accuracy

For initialization, the algorithm does the following:

- Let the data set be *D*, initialize k empty sets A_1, A_2, ... A_k
- Find the pair with minimum seperation in *D*, remove it from *D* and insert in A_1.
- Find the point in D closest to set A_1 using any distance metric, remove it from D and insert it in A_1.
- Keep repeating previous step until the size of A_1 reaches a threshold, preferably 0.75×(n/k).
- Perform steps 2-4 for sets A_2, A_3,..... A_k
- Centroids of A_1, A_2, ... A_k are the initial centroids

Similar to Agglomerative Clustering! The algorithm forms k clusters by repeatedly performing agglomerative clustering on data set and removing the cluster points from the main data set.

A major drawback of this algorithm is it's time complexity. To find the closest pair the algorithm has to perform O(n^2) operations, where n is the length of data set. Our previous algorithm had the order O(n × k × i), where k and i are number of clusters and iterations respectively. This means our previous 5 clustered cat photo would take approximately (17.9 × 256 × 256)/(5 × 5) seconds = 13 hours!! Hence this cannot be incorporated in this project.

Improving Efficiency

The Idea:

- For each data point, store the distance to the nearest cluster.
- In the next iteration, compute the distance to the previous nearest cluster.
- If the new distance is less than or equal to the previous distance, the point stays in its cluster
- No need to compute its distances to the other cluster centers.
- Saves time required to compute distances to k−1 cluster centers.

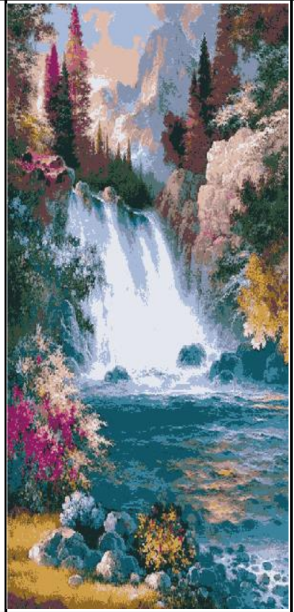
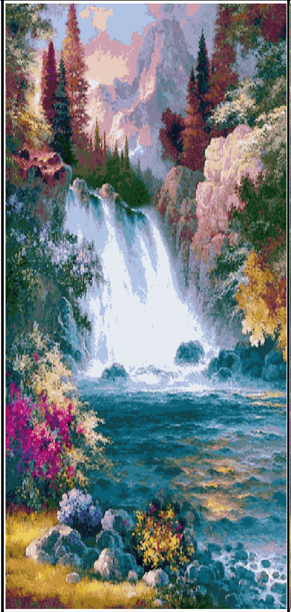
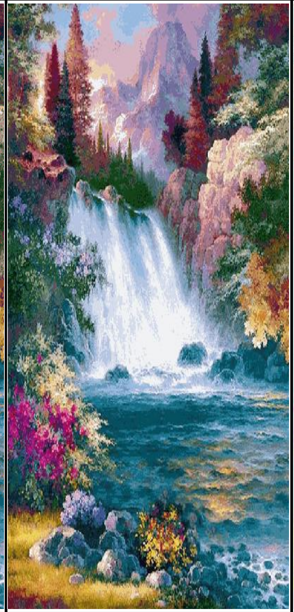
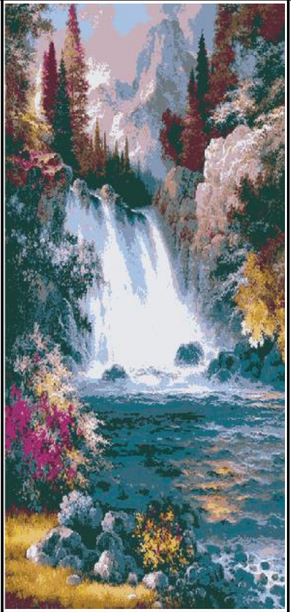
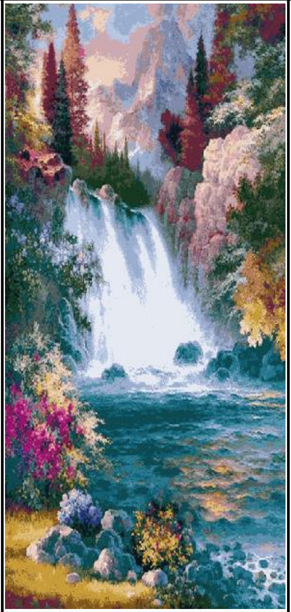
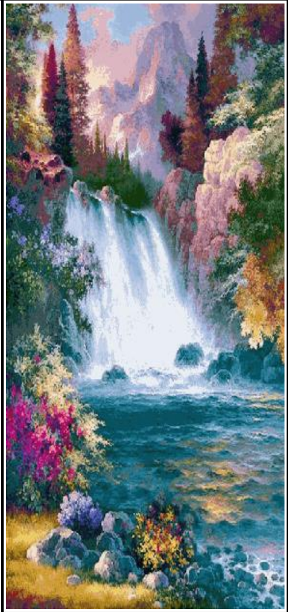
Implementation of Enhanced k-means:

- Modify the assign function of the standard k-means.
- The function first checks the distance of each point from centroid of previously assigned cluster.
- If it is less than or equal to the distance from the previously assigned centroid then the cluster assigned remains unchanged. Otherwise, it computes distance of data point from each centroid just like the assign function of standard algorithm.

Comparing Algorithms

Both algorithms are run on this 592×800 dimensional image with 473600 pixels by varying the number of clusters

Comparing Results

Algorithm	Standard K-means			Enhanced K-means		
Image						
Clusters	20	40	80	20	40	80
Time	5 min 44 sec	13 min 39 sec	24 min 41 sec	3 min 33 sec	7 min 25 sec	14 min 20 sec

References

1. Research paper for Enhanced K-means algorithm:
   (PDF) Improving the Accuracy and Efficiency of the k-means Clustering Algorithm

1. Research paper for justification of idea behind enhancement:
   www.jzus.zju.edu.cn/oldversion/opentxt.php?doi=10.1631/jzus.2006.A1626

THANK YOU ☺