COLOR REDUCER

Prasad Girish Chitnis
Pgc5277@rit.edu

Pratik Shirish Kulkarni
Psk7534@rit.edu

***Abstract*—Color reducer is an approach used to reduce number of colors in an image to some specific value. An image can be treated as an array of pixels with each pixel represented as a value corresponding to color of that pixel. These values can be in the range of 0-255. Using color reducer each pixel will be mapped to a cluster and a pre-determined value of color will be assigned to that pixel resulting in reducing the number of colors in an image.**

I. INTRODUCTION

Digital images are formed by assigning a color value to a pixel. This color value is obtained by additive method using red, green and blue as basic colors. Various combinations of these colors can be used to generate different colors. As these color values can be represented using an 8-bit integer an image can have 256 colors. In color reducer we are limiting the number of colors in an image to 5 colors. This is achieved by using clustering technique to assign each pixel a suitable cluster. Each cluster can be assigned a specific color value selected from pre-determined set of 5 colors.
This way we can we reduce the range of possible color values for a pixel from 0 – 255 to 0 – 4. Re-drawing the image using new pixel values we render image with reduced colors.

II. DETERMINING TARGET COLORS

A digital image is made of pixels with 2^8 possible colors for each pixel. Hence to select 5 colors we referred few images and most prominent colors were found to be Red , Blue ,Green , Yellow , Black , White , Grey, Orange , Pink , Purple and Maroon. So to decide which 5 colors we should use we divided the colors into 3 groups of 5

Group 1: Black, White, Red, Blue, Green

Group 2: Yellow, Orange, Pink, Black, White

Group 3: Purple, Maroon, Black, White, Grey.

So then to finalize the color group we implemented these groups on various images. And after comparing the results what we saw was that the group 1 provided with the best contrast when compared with the other two groups.

III. NATURE OF THE PROBLEM

Classification is a supervised learning technique in which you have an already predefined set of classes and you just want to know which class the new dataset belongs to. Whereas clustering is an unsupervised learning technique where you are given a dataset and in which starting from scratch you have to find whether there is any relation between the objects in the data set. So for this problem we decided to define it as a clustering problem rather than classification problem as we did not have any predefined relationships.

Now after deciding to tackle the problem as a Clustering problem we had various algorithms that we could choose. So after comparing KMeans, DBscan, Hierarchical clustering and Fuzzy clustering Algorithm we found out that KMeans Algorithm was easier to understand, robust, fast and efficient as compared to others. It also had advantage over others as it is comparatively fast and uses less memory. KMeans Algorithm measures the color closeness using Euclidian distance formula. A pixel is assigned to a cluster by comparing its distance to each cluster centroid and nearest match (cluster having least distance) is selected.

IV. DATA PREPARATION

As we are using Weka's SimpleKMeans object to build the clusterer, we can't use the image as it is. To solve this problem we had couple of approaches. We first tried writing a special Reader that will read image as an array of Pixels and then provide red, green and blue components to Instances object. But this Reader failed to achieve specifications required by Instances. Hence we decided to make a ARFF file from the image and then feed it to Instances to read the ARFF file. For this we used MakeNegative.java from MediaLibrary as a reference. To extract pixel information from the image we used getPixel method from Picture class which returns an array of Pixel. Using this we can reference individual pixel and get red, green, blue components. These values are then written to a ARFF file.

To read this data file we are using a regular BufferedReader which can be passed to an instance of Instances. Instances object is used by KMeans clusterer to read data.

V. ALGORITHM

Steps:

DATA EXTRACTION:

- Take image file as input from the user.
- Extract the pixels from the Image file.
- Extract information about every pixel from the image i.e. RGB value

DATA PREPARATION:

- After extracting information from the Image the format in which the data was stored was not usable for the Clustering Algorithm that we used so to tackle that problem we had to Store the data in an ARFF file.

CLUSTERING:

- Create Object of the Class Instances which will be used to read the data from the ARFF file.
- As we are using K-means Algorithm Create the Object of class SImpleKMean. For the Object of Class SImpleKMean set parameters like Preference Order, number of Clusters etc. as per your test case.
- Read data from the ARFF file and build the Cluster.

INTERPRETING OUTPUT AND BUILDING THE NEW IMAGE.

- After Calculating the Clusters assign every cluster a specific color from our selected colors.
- Now traversing through every pixel check which cluster the pixel belongs to and then assign the color that was given to the specific cluster to the pixel.

VI. RESULTS

As explained in Selection of Target Colors we compared the result of the output received from the various color groups and then based on the contrast of the image we decided to go forward with { Black , White , Red , Blue , Green} . As shown from images A i.e. is the original image and image C (image in which clustering is applied) we can see the reduction to the 5 colors we selected. Image B is what we get when we do not apply our target color to the output. The only time when our solution fails is when we apply it on a binary image. Consider Image D and E. So when we implemented our solution on Image D we were only able to find 2 clusters instead of the required 5. So when we get output we see the same Image

IMAGE A

Image B

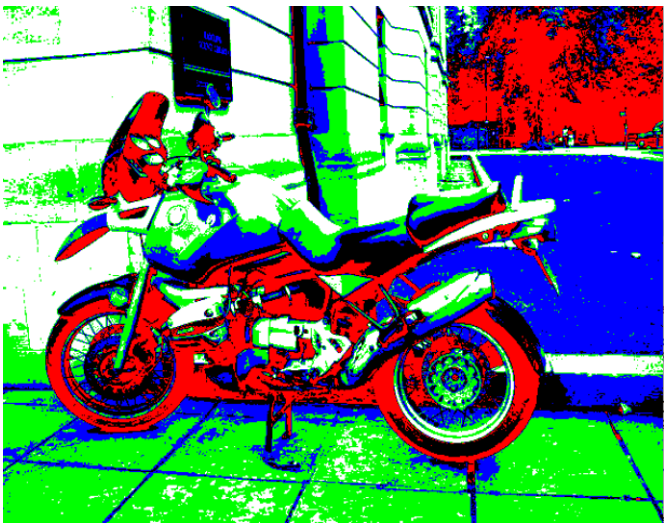

Image C

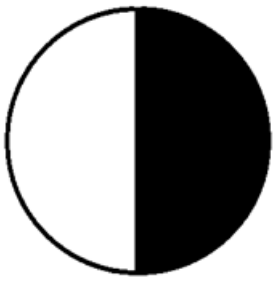

Image D

VII. REFERENCES

[1] Sandra Sagaya Mary. D. A, Tamil Selvi. R, "A Study of K-Means and Cure Clustering Algorithms," International Journal of Engineering Research & Technology , Vol. 3 - Issue 2 (February - 2014)

[2] Link: http://weka.sourceforge.net/doc.stable/overview-summary.html