RGB HISTOGRAM BASED CLUSTERING

Bakhtiyar Abbasov
Department of Computer Engineering
Yildiz Technical University, 34220 Istanbul, Türkiye

bakhtiyar.abbasov@std.yildiz.edu.tr

***Abstract*—This article delves into the application of K-means clustering on image data utilizing RGB histograms. The study investigates the challenges encountered in effectively categorizing images based on their color distributions. Python, along with key libraries such as NumPy, Pandas, Scikit-Learn, Seaborn, and Matplotlib, is employed in Google Colab environment to implement and anlayze K-means clustering techniques. Notably, the initial results showcase a highly successful system, achieving an overall accuracy of well above 90%. The study utilizes a confusion matrix to visualize the results and compares the manual K-means implementation with the Scikit-learn library's function, providing valuable insights into the complexities of image analysis and clustering methodologies. These findings shed light on the complexities inherent in image analysis and clustering methodologies, underscoring the need for further exploration and refinement in this evolving field.**

***Keywords*—RGB, K-Means, clustering, RGB, HSV, histogram, class, label, vector distance, centroid, accuracy, confusion matrix.**

I. Introduction

This study represents a deep exploration into the realm of image clustering methodologies, leveraging a tailored K-means algorithm for a nuanced examination. A curated dataset of 100 diverse images, meticulously categorized across five distinct color classes, forms the core of this investigation into image clustering based on color histograms. Each class encompasses a meticulously chosen set of 20 images, encompassing a broad spectrum of color variations. The findings present an encouraging picture, revealing exceptional success rates. The clustering process, executed over ten iterations, showcased remarkably high accuracy scores ranging between 90% and 97%, underscoring the effectiveness of this custom K-means approach. While celebrating these commendable outcomes, this research also aims to delve into the subtle complexities and challenges inherent in the clustering process, paving the way for further refinement and exploration in this domain.

II. System Design

A. Dataset Preparation

The dataset utilized for this project comprises 100 images, each with dimensions of 416x416 pixels, distributed across five color classes: red, green, blue, gray, and white. This dataset was accessed via Roboflow[1]. To accommodate the Google Colab environment used in this project, a zip file containing these images was provided to the system, which, upon extraction, organized the images into five respective color-labeled folders. To facilitate data handling, a helper function was employed, aiding in the conversion of the images into a three-dimensional array, thereby separating the color values (R, G, B) within each image. Subsequently, another function from the NumPy library was used to extract the histograms for R, G, and B colors, resulting in 256-element arrays that represent the color distribution within each image. After processing the 100 images, three arrays—each containing 100x256 dimensions representing R, G, and B histograms—were created. These arrays were then concatenated along the y-axis, generating a single 100x768 array. Each row in this array represents the stacked histograms for the R, G, and B values of a specific image.

B. Clustering Methodology

The clustering process begins by converting the obtained feature vectors into a DataFrame. The DataFrame structure facilitates easy data manipulation and analysis. This step involves transforming the raw data into a structured format, enhancing its accessibility and comprehensibility for further processing. The clustering algorithm is applied to the DataFrame, where each row represents a data point and each column corresponds to a feature. The algorithm initializes random cluster centers and iteratively assigns data points to the nearest cluster center based on their similarity or distance metric. During each iteration, the algorithm computes the similarity between data points and cluster centers, identifying the closest cluster center for each data point. Subsequently, it reallocates data points to clusters based on these calculated distances, aiming to minimize intra-cluster differences and maximize inter-cluster dissimilarities. The iterative process continues until convergence, where data points remain consistently assigned to the nearest cluster center across successive iterations. The result is an array representing predicted cluster labels for each data point, indicating their respective cluster memberships based on the implemented clustering algorithm.

C. Analysis of Results:

1) Overall Accuracy Assessment:: To quantify the overall accuracy of the clustering process, standard evaluation metrics such as accuracy scores are calculated. These metrics provide a global perspective on the performance of the clustering algorithm. Using both library-based and manually implemented K-means algorithms, the accuracy scores are derived to ascertain the effectiveness of the methodologies employed.

2) Class-specific Accuracy:: A deeper analysis involves computing class-specific accuracies to understand the performance of the clustering algorithm for each color class. Utilizing masks based on color labels, individual accuracy scores are determined for each class.

3) 3. Confusion Matrix:: A pivotal visual tool in result analysis, the confusion matrix, is employed to showcase the predicted labels against the actual labels. This matrix presents a clear illustration of the clustering performance across different color classes, facilitating a detailed evaluation of misclassifications and the clustering accuracy.

III. Experimental Results

The K-means clustering was initiated with the selection of five random points as centroids. This process was iterated ten times to ensure robustness and reliability. Table 1 showcases the overall accuracy and class-specific accuracy attained from each iteration, demonstrating the consistency and effectiveness of the clustering process.

Total Accuracy	Blue	Gray	Green	Red	White
94.00%	100.00%	95.00%	90.00%	90.00%	95.00%
93.00%	90.00%	85.00%	90.00%	100.00%	100.00%
91.00%	95.00%	85.00%	85.00%	90.00%	100.00%
96.00%	100.00%	95.00%	95.00%	95.00%	95.00%
90.00%	90.00%	85.00%	85.00%	90.00%	100.00%
98.00%	95.00%	95.00%	100.00%	100.00%	100.00%
92.00%	90.00%	85.00%	90.00%	95.00%	100.00%
93.00%	100.00%	90.00%	85.00%	90.00%	100.00%
97.00%	100.00%	90.00%	85.00%	90.00%	100.00%
90.00%	100.00%	90.00%	85.00%	85.00%	90.00%

Table 1 Total And Class Accuracy Percentages

Notably, the confusion matrix displayed in Figure 1 illustrates the high success rate of the clustering model, highlighting its proficiency in accurately classifying images into their respective color categories.

Subsequently, Figure 2 portrays a curated selection of images, showcasing five correctly and one incorrectly assigned image for each color class. This visualization provides a closer examination of the clustering model's performance for individual classes, shedding light on the instances of accurate and erroneous classifications.

Moreover, Figure 3 provides a focused view, displaying one mismatched image per class. This specific representation offers a deeper understanding of the instances where the clustering algorithm encountered challenges in accurately categorizing certain images within their respective color classes.

IV. Conclusion

In summary, the implemented system showcased commendable accuracy, surpassing the overall 90% mark. However, challenges were observed, particularly concerning color tone variations that impacted classifications, notably within the gray and white categories. Exploring alternative color spaces, such as HSV histograms, may fortify the model's resilience to such complexities. Despite its current limitations, the system shows promise for future enhancements and potential applications in similar studies.

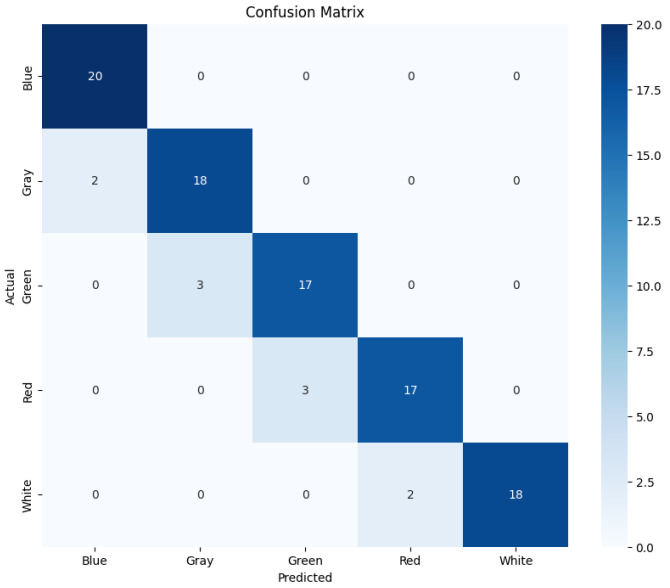

Figure 1 Confusion Matrix

Figure 2 Correctly Predicted Images

Figure 3 Incorrectly Predicted Images

References

[1] example01, "Roboflow." [Online]. Available: https://universe.roboflow.com/example01/color-oquyt