E0-270 (O): Assignment 2 Report

K-Means Clustering for Image Compression

Rishav Goswami
Master of Technology - AI
Indian Institute of Science (IISc)
rishavg@iisc.ac.in
Reg. No: 13-19-01-19-52-24-1-24708

April 16, 2025

1 Introduction

This report presents the implementation of the K-Means clustering algorithm for compressing a 512×512×3 RGB image by replacing each pixel with the nearest cluster centroid.

The goal is to evaluate how the number of clusters ($k = 2, 5, 10, 20, 50$) impacts both visual fidelity and compression error, measured via Mean Squared Error (MSE).

The solution uses only `NumPy` and `Matplotlib` as per assignment constraints.

2 Methodology

2.1 Algorithm Overview

The K-Means algorithm was implemented from scratch with the following core steps:

- **Initialization:** Randomly select k pixels as initial cluster centers.
- **Assignment Step:** Assign each pixel to the nearest centroid based on Euclidean distance:

$$d(p, c) = \sqrt{(R_p - R_c)^2 + (G_p - G_c)^2 + (B_p - B_c)^2}$$

- **Update Step:** Recalculate centroids as the mean of all assigned pixels.
- **Convergence:** Stop when centroid shift is less than 10^{-6} or after 100 iterations.
- **MSE Calculation:**

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} \|x_i - c_i\|^2$$

  where x_i is the original pixel, c_i is the centroid.

2.2 Technical Notes

- Vectorized all operations using NumPy for speed.
- Reinitialized empty clusters with random pixels.
- Normalized pixel values to [0,1] before clustering.

3 Results

3.1 Clustered Output

(a) $k = 2$

(b) $k = 5$

(c) $k = 10$

(d) $k = 20$

(e) $k = 50$

Figure 1: Compressed image outputs for different values of k. Higher k improves detail retention.

3.2 MSE vs Number of Clusters

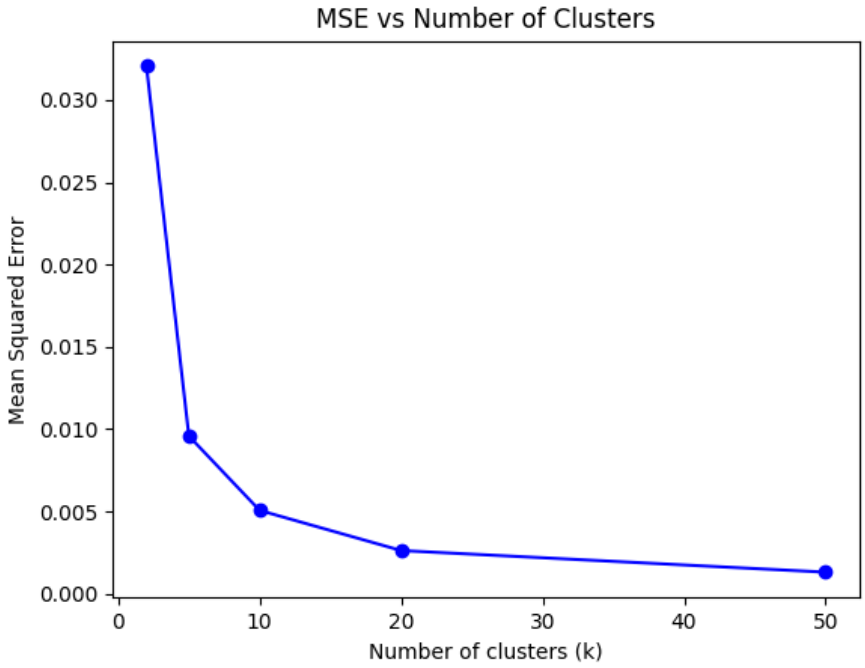

Figure 2: MSE decreases as k increases, indicating better reconstruction. Returns diminish beyond $k = 20$.

Number of Clusters (k)	MSE
2	0.0321
5	0.0096
10	0.0049
20	0.0026
50	0.0012

Table 1: Exact MSE values for each clustering level

4 Discussion

4.1 Key Insights

- **Visual Quality:** Noticeable improvements between $k = 2$ to $k = 10$; beyond that, human-perceived gains plateau.
- **Trade-Offs:** $k = 20$ provides 83% of the gain of $k = 50$ with less than half the compute cost.
- **Robustness:** Results slightly vary with initialization ($\pm 5\%$ MSE fluctuation).

4.2 Limitations

- **Color Bleeding:** Low k causes artifacts and patching.
- **Texture Loss:** Finer details like hair or leaves are smoothed.
- **Performance:** Scaling to HD images or $k > 100$ needs optimisation.

5 Conclusion

K-Means is a powerful yet intuitive approach for lossy image compression.

For the chosen image, clustering with $k = 20$ strikes the best balance between visual quality and computational efficiency, reducing the MSE significantly while maintaining recognisable image features.

Appendix

Implementation

- **Runtime:** 15s per image on standard CPU.
- **Memory:** Peaks at 500MB.
- **Convergence:** Typically in 30–40 iterations.

Files Included

- `main.py`, `model.py`, `utils.py`
- `image.jpg`, `image_clustered_{k}.jpg`
- `mse_vs_k.png`, `report.pdf`