# 

# CS-351-Artificial Intelligence

# Genetic Algorithm-Based image compression

# *Muhammad Huzaifa(2022389)*

***Saadullah Haideri(2022420)***

# Genetic Algorithm-Based Image Compression

## Introduction

Image compression is very important for the reduction of storage and transmission requirements of digital images. This report describes the implementation of a sophisticated image compression algorithm with the use of Genetic Algorithms. The method takes advantage of the ability of GAs to find optimal solutions using evolutionary principles, in an efficient manner, approximating the original image through a reduced color palette while maintaining high visual fidelity.

## Objectives

* Reduce an image's size by compressing it with a limited palette of colors.
* Make sure the compressed image closely resembles the original in terms of color quality and detail.
* Minimize computational overhead while achieving high-quality results.
* Present the key metrics: reconstruction error and compression efficiency.

## Code Workflow

### 1. Image Loading and Preprocessing

The user uploads an image, which is then resized and normalized in order to prepare it for processing. This step ensures uniformity and compatibility with the algorithm.

**Key Functions:**

* upload\_image: Allow user-uploaded files.
* load\_image: Resizes the image and normalizes pixel values.

### 2. Flattening the Image

It creates a two-dimensional array in which each row represents a pixel with its RGB values. This representation makes operations like clustering and fitness evaluation much easier to handle.

**Function:**

* flatten\_image: Flattens the image into a pixel array and stores its original shape for reconstruction.

### 3. Genetic Algorithm Operations

The core of the algorithm lies in its evolutionary operations:

#### Initialization:

An initial population of color palettes is created by using the KMeans clustering algorithm in order to produce meaningful starting points.

**Function:**

* initialize\_population: Generates palettes around the image's color clusters.

#### Selection:

Parents are selected according to their fitness values, which are inversely proportional to the reconstruction error.

**Function:**

* select\_parents: Retains the top-performing palettes for reproduction.

#### Crossover:

New palettes are created by splicing together chunks of parent palettes.

**Function:**

* crossover: Creates offspring by combining parent palettes.

#### Mutation:

A small, random change is introduced in the offspring palettes to maintain diversity and refine solutions.

**Function:**

* mutate: Applies subtle changes to palette colors.

### 4. Fitness Evaluation

Fitness is determined by comparing the reconstructed image against the original using the mean squared error (MSE).

**Function:**

* calculate\_fitness: Evaluates the quality of a palette based on reconstruction error.

### 5. Reconstruction and Visualization

Best palette is applied to reconstruct the compressed image. Results are presented with metrics like reconstruction error, image size, and execution time.

**Functions:**

* reconstruct\_image: Reconstructs the image from the best palette.
* plot\_results: Plots the original and compressed images, fitness evolution, and important metrics.

## Results

The Genetic Algorithm achieves high-quality image compression with improved color fidelity and reduced pixelation. Using KMeans initialization and fine-tuned mutation rates, the algorithm produces a compressed image that very closely resembles the original.



### Metrics:

* **Original Image Format:** Original Image Format: JPEG
* **Original Image Size:** 192.00 KB.
* **Compressed Image Size:** 64.19 KB
* **Reconstruction Error (MSE):** 0.000206
* **Execution Time:** 1289.96 seconds.

## Key Improvements

* **Color Fidelity:** Larger palette size and improved initialization provide better color representation.
* **Reduced Pixelation:** Higher-quality reconstruction ensures smooth transitions between colors.
* **Enhanced Efficiency:** Streamlined operations and controlled mutation reduce computational overhead.

## Conclusion

The Genetic Algorithm-based image compression method effectively reduces image size while maintaining high visual quality. By incorporating advanced initialization techniques and precise genetic operations, the algorithm demonstrates robustness and efficiency, making it a valuable tool for digital image processing. Future enhancements could include adaptive palette sizes and parallel processing to further improve performance.

## Recommendations

* **Experiment with Larger Images:** Test the algorithm on higher-resolution images to evaluate scalability.
* **Optimize Execution Time:** Incorporate parallel computing to accelerate convergence.
* **Integrate Adaptive Techniques:** Dynamically adjust parameters such as palette size and mutation rate based on image complexity.