**Code Explanation:**

**Libraries Used:**

* **OpenCV (cv2)**: Used for reading, resizing, and manipulating images.
  + To install: **pip install opencv-python**
* **NumPy (numpy)**: Used for numerical operations and array manipulations.
  + To install: **pip install numpy**
* **scikit-image (skimage)**: Specifically, the **structural\_similarity** module to calculate the Structural Similarity Index (SSI) between images.
  + To install: **pip install scikit-image**
* **Matplotlib (matplotlib)**: Used for plotting and visualizing the comparison results.
  + To install: **pip install matplotlib**

**Workflow:**

1. **Import Libraries**: Import necessary libraries - **cv2**, **numpy**, **structural\_similarity** from **skimage.metrics**, and **matplotlib.pyplot**.
2. **File Paths**: Define paths to the base image and folder containing testing images.
3. **Read Base Image**: Read the base image using OpenCV (**cv2.imread()**). Check if the image is successfully loaded.
4. **Resize Base Image**: Resize the base image to a common size if it's not already of the desired size.
5. **Convert Base Image to Grayscale**: Convert the resized base image to grayscale for comparison (**cv2.cvtColor()**).
6. **Comparison Initialization**: Initialize lists to store comparison results.
7. **SSIM Calculation for Testing Images**:
   * Loop through the testing images folder (assuming names like Testimage1.jpg, Testimage2.jpg, ..., Testimage20.jpg).
   * Read each testing image using OpenCV (**cv2.imread()**). Check if the image is successfully loaded.
   * Resize testing images to match the size of the base image.
   * Convert testing images to grayscale for comparison.
   * Calculate Structural Similarity Index (SSI) using **ssim()** from **skimage.metrics**.
   * Calculate the similarity percentage based on the SSI.
   * Store comparison results (similarity percentages and corresponding images).
8. **Display Comparison Results**:
   * Plot all comparisons in a single window using Matplotlib (**matplotlib.pyplot.imshow()**).
   * Display the testing images alongside their similarity percentages relative to the base image.

Installation Instructions:

To install the required libraries, use the following commands in your command prompt or terminal:

* OpenCV: **pip install opencv-python**
* NumPy: **pip install numpy**
* scikit-image: **pip install scikit-image**
* Matplotlib: **pip install matplotlib**

**Code Explanation:**

import cv2

import numpy as np

from skimage.metrics

import structural\_similarity as ssim

import matplotlib.pyplot as plt

* **import statements:** These lines import necessary libraries (**cv2**, **numpy**, **structural\_similarity** from **skimage.metrics**, and **matplotlib.pyplot**) for image processing, numerical computations, SSIM calculation, and visualization.

base\_image\_path = 'C:\\Users\\Muhammad Kamran\\Desktop\\CVA1\\Baseimage.jpg' testing\_images\_folder = 'C:\\Users\\Muhammad Kamran\\Desktop\\CVA1\\Testing Images\\'

* **File Paths:** Define the paths to the base image and the folder containing testing images.

base\_image = cv2.imread(base\_image\_path)

* **Read Base Image:** Read the base image using OpenCV's **imread** function.

if base\_image is None:

print("Error: Unable to read the base image.")

else:

# Resize base image to a common size if necessary

width = 200 # Set a common width for resizing

height = 200 # Set a common height for resizing

base\_image\_resized = cv2.resize(base\_image, (width, height))

base\_gray = cv2.cvtColor(base\_image\_resized, cv2.COLOR\_BGR2GRAY)

* **Base Image Processing:** Checks if the base image is read successfully. If so, resizes it to a common size and converts it to grayscale.

similarity\_scores = []

compared\_images = []

for i in range(1, 21): # Loop through 20 testing images (Testimage1.jpg to Testimage20.jpg) testing\_image\_path = testing\_images\_folder + f'Testimage{i}.jpg'

testing\_image = cv2.imread(testing\_image\_path)

# Processing each testing image similarly as the base image

# ... (code for resizing, converting to grayscale, calculating SSIM, and storing results)

* **Loop through Testing Images:** Reads each testing image, resizes it, converts to grayscale, calculates SSIM against the base image, and stores the results.

plt.figure(figsize=(15, 10))

columns = 5 # Number of columns in the grid

rows = 4 # Number of rows in the grid

for i in range(len(compared\_images)):

# Plotting the compared images in a grid format

# ... (code for displaying the images with similarity scores)

* **Display Comparison Results:** Utilizes Matplotlib to create a visualization showing the base image compared to each testing image with their similarity scores.

**Libraries Installation:**

You can install the required libraries via pip in your command prompt or terminal:

pip install opencv-python numpy scikit-image matplotlib