# Import necessary libraries  
import numpy as np  
from skimage import io  
from skimage.color import rgb2lab  
from skimage.measure import shannon\_entropy  
from colormath.color\_diff import delta\_e\_cie2000  
from colormath.color\_objects import LabColor  
  
  
# Function to calculate CIEDE2000 color difference  
def calculate\_color\_difference(color1, color2):  
 lab\_color1 = LabColor(color1[0], color1[1], color1[2])  
 lab\_color2 = LabColor(color2[0], color2[1], color2[2])  
 return delta\_e\_cie2000(lab\_color1, lab\_color2)  
  
  
# Function to calculate fractal dimension  
def calculate\_fractal\_dimension(image):  
 # Perform fractal analysis (example implementation)  
 # This function can be replaced with more advanced fractal analysis algorithms  
 return shannon\_entropy(image)  
  
  
# Function to calculate overall image quality score  
def calculate\_image\_quality(image\_path):  
 # Load image  
 image = io.imread(image\_path)  
  
 # Convert image to Lab color space  
 lab\_image = rgb2lab(image)  
  
 # Calculate average color difference using CIEDE2000  
 color\_difference\_sum = 0  
 for i in range(lab\_image.shape[0]):  
 for j in range(lab\_image.shape[1]):  
 color\_difference\_sum += calculate\_color\_difference(lab\_image[0][0], lab\_image[i][j])  
 average\_color\_difference = color\_difference\_sum / (lab\_image.shape[0] \* lab\_image.shape[1])  
  
 # Calculate fractal dimension  
 fractal\_dimension = calculate\_fractal\_dimension(image)  
  
 # Combine results into overall quality score (simple linear combination)  
 quality\_score = 0.6 \* average\_color\_difference + 0.4 \* fractal\_dimension  
  
 return quality\_score  
  
  
# Path to the image  
image\_path = 'example\_image.jpg'  
  
# Calculate image quality score  
quality\_score = calculate\_image\_quality(image\_path)  
  
# Output image quality score  
print("Image Quality Score:", quality\_score)