**Generating the Dataset (Real and Fake Images)**

import cv2

import numpy as np

import os

from skimage.metrics import structural\_similarity as ssim

import pandas as pd

# Create directories for real and fake images

if not os.path.exists('real\_images'):

os.makedirs('real\_images')

if not os.path.exists('fake\_colorized\_images'):

os.makedirs('fake\_colorized\_images')

# Function to generate fake colorized images and calculate metrics

def generate\_images\_and\_metrics(num\_real\_images=100, num\_fake\_images=100):

data = {

'Image\_ID': [],

'Image\_Type': [],

'Color\_Consistency\_SSIM': [],

'Color\_Difference\_∆E': [],

'File\_Path': []

}

for i in range(1, num\_real\_images + 1):

# Generate or load a real image (placeholder here with random array)

real\_image = np.random.randint(0, 256, (224, 224, 3), dtype=np.uint8)

real\_image\_path = f'real\_images/real\_image\_{i}.jpg'

cv2.imwrite(real\_image\_path, real\_image)

# Save metadata for real image

data['Image\_ID'].append(i)

data['Image\_Type'].append('Real')

data['Color\_Consistency\_SSIM'].append(1.0) # SSIM for real image is perfect

data['Color\_Difference\_∆E'].append(0) # No color difference in real images

data['File\_Path'].append(real\_image\_path)

for i in range(1, num\_fake\_images + 1):

# Generate a grayscale image and convert it to a fake colorized version

fake\_image\_gray = np.random.randint(0, 256, (224, 224), dtype=np.uint8)

fake\_colorized\_image = cv2.applyColorMap(fake\_image\_gray, cv2.COLORMAP\_JET)

fake\_image\_path = f'fake\_colorized\_images/fake\_image\_{i}.jpg'

cv2.imwrite(fake\_image\_path, fake\_colorized\_image)

# Calculate SSIM between fake colorized and grayscale (as a simple example)

fake\_image\_ssim = ssim(fake\_image\_gray, cv2.cvtColor(fake\_colorized\_image, cv2.COLOR\_BGR2GRAY))

# ∆E color difference (dummy example here)

color\_difference = np.random.uniform(20, 40)

# Save metadata for fake image

data['Image\_ID'].append(num\_real\_images + i)

data['Image\_Type'].append('Fake')

data['Color\_Consistency\_SSIM'].append(fake\_image\_ssim)

data['Color\_Difference\_∆E'].append(color\_difference)

data['File\_Path'].append(fake\_image\_path)

# Create a DataFrame

df = pd.DataFrame(data)

return df

# Generate dataset

df = generate\_images\_and\_metrics()

# Save dataset to Excel

output\_path = 'fake\_colorized\_image\_detection\_dataset.xlsx'

df.to\_excel(output\_path, index=False)

print(f"Dataset saved to {output\_path}")