**ALGORITHMS**

**Principal component analysis (PCA):**

* It is a dimensionality reduction method that is often used to reduce the dimensionality of large data sets.
* By transforming a large set of variables into a smaller one that still contains most of the information in the large set.

PCA can be used to visualize high-dimensional data in two or three dimensions.

import os

import random

from PIL import Image

import numpy as np

from sklearn.decomposition import PCA

input\_folder = r'D:\project core\original data sets'

output\_folder = r'D:\project core\reduced output'

target\_size = (2296, 1724) # Target size for resized images

batch\_size = 20 # Adjust based on available memory

# Ensure the output directory exists, create it if not

os.makedirs(output\_folder, exist\_ok=True)

# Load images and convert them to numpy arrays

images\_data = []

for file in os.listdir(input\_folder):

input\_path = os.path.join(input\_folder, file)

try:

with Image.open(input\_path) as img:

resized\_img = img.resize(target\_size)

img\_array = np.array(resized\_img).reshape(-1) # Flatten the 2D array into 1D

images\_data.append(img\_array)

except Exception as e:

print(f"Error processing {file}: {str(e)}")

# Randomly subsample data if it's too large

subsample\_size = 1000 # Adjust based on your needs

if len(images\_data) > subsample\_size:

images\_data = random.sample(images\_data, subsample\_size)

# Convert the list of image arrays to a 2D numpy array

images\_data = np.array(images\_data)

# Use PCA for dimensionality reduction

pca = PCA(n\_components=2)

pca\_results = pca.fit\_transform(images\_data)

# Visualize or save PCA results (as an example, you might want to visualize or analyze these results)

import matplotlib.pyplot as plt

plt.scatter(pca\_results[:, 0], pca\_results[:, 1])

plt.show()

# Save PCA results (as an example, you might want to visualize or analyze these results)

for i, file in enumerate(os.listdir(input\_folder)):

output\_path = os.path.join(output\_folder, file.replace('.jpg', '\_pca.jpg'))

x, y = pca\_results[i]

# Create a visualization showing the PCA points

pca\_img = Image.new('RGB', (500, 500))

pca\_img.putpixel((int(x) + 250, int(y) + 250), (255, 0, 0)) # Red dot for PCA point

pca\_img.save(output\_path)

print(f"Saved PCA visualization for {file} to {output\_path}")