import os

import cv2

import numpy as np

import matplotlib.pyplot as plt

from keras.models import load\_model

from sklearn.neighbors import NearestNeighbors

from bow import BoW

import preprocessing

def append\_train\_ds(path, save\_path='codes', model=load\_model('encoded.h5'), bow\_model = BoW(code\_size=256, n\_clusters=100, centroids\_path='kmeans\_centroids.npy')):

folder, file = os.path.split(path)

img = cv2.imread(path)

blocks = preprocessing.preprocess(img, block\_shape=(64, 64))

codes = model.predict(blocks, batch\_size=16)

codes = np.reshape(codes, [-1, 256])

bow = bow\_model.transform(codes)

img\_idx = len(os.listdir(save\_path))

np.save(save\_path + '/' + str(img\_idx), bow)

def infer(img, n\_neighbors=10, data\_folder='codes', model=load\_model('encoded.h5'), bow\_model = BoW(code\_size=256, n\_clusters=100, centroids\_path='kmeans\_centroids.npy'), visualize\_best=True):

# initialize

if type(img) == str:

img = cv2.imread(img)

knn = NearestNeighbors(n\_neighbors)

# read & fit data

files = os.listdir(data\_folder)

data = []

for file in files:

data.append(np.load(data\_folder + '/' + file))

data = np.vstack(data)

knn.fit(data)

# transform data

blocks = preprocessing.preprocess(img, block\_shape=(64, 64))

codes = model.predict(blocks, batch\_size=16)

codes = np.reshape(codes, [-1, 256])

bow = bow\_model.transform(codes)

bow = np.expand\_dims(bow, axis=0)

nearest = knn.kneighbors(bow, return\_distance=False)[0]

return nearest