Homework2 , CS 682 Spring20
S M Hasan Mansur

- I have posted the results and scripts to http://mason.gmu.edu/~smansur4/cs682hw2/
- Login credentials:
  Username: testuser
  Password: testpass
- Scripts also submitted in Blackboard along with this report.

Problem 1

(1) To browse & load an image(.png), I used a dialog box based on tkinter. The image was displayed using the cv2.imshow() method. I also tried the same exercise with .jpeg images. As jpeg is a lossy compression scheme, I observed that the intensity of a pixel is less than the same pixel of the .png format.

(2) I used the cv2.calcHist() method in a loop to create histogram for each channel (B,G,R) & finally plotted them using matplotlib.

(3) To detect the mouse cursor movement on the image, I used mouse event cv2.cv2.EVENT_MOUSEMOVE. To draw a square 11x11 window (actually 13 × 13 with 1 pixel border) I used the method cv2.rectangle(). Whenever the cursor moves, the square window also moves as well, having the cursor position at its center. To display the information, I created a white image full of the value 255 by using np.full() method & displayed it in a separate frame. Following information is displayed in that separate window:

- Coordinates (x,y) of the reference point (p)
- RGB values at p
- Intensity value at p (average of RGB values at p)
- Mean and standard deviation of the window, Wp. For this, I used the cv2.meanStdDev() method.

(4) I observed several windows while moving the reference point(p) on the image & it is noticeable that if the standard deviation is 0, then the window mean for each channel is equal to that of the corresponding channel value at reference point or pixel (p). This represents the homogeneous distributions of image values. On the other hand, windows show inhomogeneous distribution where the standard deviation > 0 and the mean value deviates from that of the reference point's values. Also from histogram we can notice that in case of homogeneous distributions, majority of the pixels hold similar value whereas this is not true for inhomogeneous distribution.

Problem2

(A) I used glob.glob() to read image files from ST2MainHall4 directory & load them in a list. Every file was read by cv2.imread(), separated into different channel (BGR) matrices by using slicing operation. Then, bit shift operation was executed on every channel matrix. Finally, the shifted channel matrices were merged & np.histogram() method was used to calculate histogram for that particular image. Histogram for each image was appended in a list so that it could be later used for histogram intersection & chi square measure.

(B) For histogram comparison, I wrote two functions histogram_intersection() & histogram_chi2(). For implementing intersection, I used the methods: np.true_divide(), np.sum(), np.minimum(), np.maximum(). For implementing chi square, I used the methods: np.nansum(), np.true_divide(), np.square(), np.subtract(), np.add().

(C) To compare all image pairs, I utilized the functions developed in the previous section (B). As there were a total of 99 images, the results were kept in two 99x99 matrices. Later the values in the matrices scaled between 0 to 255. Finally, I plotted the results using matplotlib. In case of intersection, large values correspond to high similarity and for chi-squared measure, low values correspond to high similarity.

References:

1. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html
2. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_histograms/py_histogram_begins/py_histogram_begins.html
3. https://data-flair.training/blogs/python-bitwise-operators/
4. https://jakevdp.github.io/PythonDataScienceHandbook/02.02-the-basics-of-numpy-arrays.html
5. https://mpatacchiola.github.io/blog/2016/11/12/the-simplest-classifier-histogram-intersection.html
6. https://matplotlib.org/gallery/images_contours_and_fields/image_annotated_heatmap.html#sphx-glr-gallery-images-contours-and-fields-image-annotated-heatmap-py
7. https://www.geeksforgeeks.org/numpy-nansum-in-python/

Scripts: Problem1

```
import cv2
import math
import numpy as np

from matplotlib import pyplot as plt
from tkinter import filedialog
from tkinter import *

def on_mouse_over(event, x, y, flags, param):
    global img
    global dashboard
    if event == cv2.cv2.EVENT_MOUSEMOVE:
        img_reset()
        cv2.rectangle(img,(x-6, y-6), (x+6, y+6),(0,0,255),1)
        intensity = sum(img[y][x])/3
        window = img[y-5:y+5, x-5:x+5]
        #print(img.shape)
        x0 = 0
        xn = img.shape[1] - 1
        y0 = 0
        yn = img.shape[0] - 1
        if (x-5 < x0 or x+5 > xn or y-5 < y0 or y+5 > yn):
            txt = "window out of boundary"
            str_mean = "mean: {}".format(txt)
            str_std = "standard deviation: {}".format(txt)
        else:
            mean, std = cv2.meanStdDev(window)
            str_mean = "window mean: " + "\n" + "R:{}, G:{},
B:{}".format(mean[2][0],mean[1][0], mean[0][0])
            str_std = "window standard deviation: " + "\n" + "R:{}, G:{},
B:{}".format(std[2][0],std[1][0], std[0][0])
        str_coordinates = "x:{}, y:{}".format(x,y)
        str_rgb = "R:{},G:{},B:{}".format(img[y][x][2], img[y][x][1], img[y][x][0])
        str_intesity = "intensity:{}".format(sum(img[y][x])/3)
        #str_mean = "mean: R:{} G:{} B:{}".format(mean[2],mean[1], mean[0])
        #str_std = "standard deviation:" + "\n" + "R:{} G:{} B:{}".format(std[2],std[1],
std[0])
```

```
        output_str = str_coordinates + "\n" + str_rgb + "\n" + str_intesity + "\n" +
str_mean + "\n" + str_std
        y0, dy = 50, 50
        for i, line in enumerate(output_str.split('\n')):
            y = y0 + i*dy
            cv2.putText(dashboard, str(line), (20, y),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 1)

def img_reset():
    global img
    global dashboard
    global filename
    #img = cv2.imread("testimage.png")
    img = cv2.imread(filename)
    dashboard = np.full((600,1200), 255, dtype='uint8')
    cv2.imshow("dashboard", dashboard)
    cv2.imshow('image', img)

def channel_histogram():
    global img
    color = ('b', 'g', 'r')
    for i, col in enumerate(color):
        histr = cv2.calcHist([img], [i], None, [256], [0, 256])
        plt.plot(histr, color = col)
        plt.xlim([0, 256])
    plt.title("Color Channels Histogram")
    plt.show(block=False)



def main():
    global img
    global dashboard
    global filename

    #img = cv2.imread("testimage.png")

    root = Tk()
    root.filename =  filedialog.askopenfilename(initialdir = "/",title = "Select
file",filetypes = (("jpeg files","*.jpg"),("png files","*.png"),("all files","*.*")))
```

```
    filename = root.filename
    img = cv2.imread(filename)

    dashboard = np.full((600,1200), 255, dtype='uint8')
    channel_histogram()
    cv2.namedWindow("image")
    cv2.setMouseCallback("image",on_mouse_over)
    while(1):
        cv2.imshow("image", img)
        cv2.imshow("dashboard", dashboard)
        k = cv2.waitKey(1) & 0xFF
        if k == 27:
            break
    cv2.destroyAllWindows()

main()
```

Scripts: Problem2

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
import glob

np.seterr(divide='ignore', invalid='ignore')

def create_histograms(files):
    histograms = []
    for file in files:
            print(file)
            img = cv2.imread(file).astype('uint16')
            b_channels, g_channels, r_channels = img[:, :, 0], img[:, :, 1], img[:, :, 2]
            img_bit_shifted = (b_channels >> 5) + ((g_channels >> 5) << 3) +
((r_channels >>5) << 6)
            img_flat = img_bit_shifted.ravel()
            histogram, bins = np.histogram(img_flat,bins=range(0,513))
            histograms.append(histogram)
    return histograms

def histogram_intersection(h1, h2):
    intersection_val = np.true_divide(np.sum(np.minimum(h1,h2)),
np.sum(np.maximum(h1,h2)))
    return intersection_val

def histogram_chi2(h1, h2):
    chi2_val = np.nansum(np.true_divide(np.square(np.subtract(h1, h2)),
np.add(h1, h2)))
    return chi2_val

def plot(plt, matrix, title):
    fig, ax = plt.subplots()
    ax.set_xticks(np.arange(0, 99, 8))
    ax.set_yticks(np.arange(0, 99, 8))
    plt.setp(ax.get_xticklabels(), rotation=90, rotation_mode="anchor")
    ax.set_title(title)
    plt.imshow(matrix)
    plt.colorbar()
    plt.show()

def main():
    hist_intersect_1 = []
    hist_intersect_2 = []
    chi2_1 = []
    chi2_2 = []

    files = [file for file in glob.glob("ST2MainHall4/" + "*")]
    files.sort()

    print("computing histograms")
    h = create_histograms(files)
    print("computing histogram intersection & chi square measure ...")
    for i in range(0,99):
        for j in range(0,99):
            hist_intersect_2.append(histogram_intersection(h[i],h[j]))
            chi2_2.append(histogram_chi2(h[i],h[j]))
        hist_intersect_1.append(hist_intersect_2)
        chi2_1.append(chi2_2)
        hist_intersect_2 = []
        chi2_2 = []
    hist_intersection = np.array(hist_intersect_1)
    chi_square = np.array(chi2_1)

    g_max = 255
    hist_intersection_scaled = g_max * hist_intersection
    chi_square_scaled = g_max * (chi_square/max(chi_square.ravel()))

    plot(plt, hist_intersection_scaled, "Histogram intersection")
    plot(plt, chi_square_scaled, "Chi-squared measure")

main()
```