calculate_histogram.py Dosyası

```python
import cv2
import numpy as np


#resize_img fonksiyonu ile gelen görsellerin boyutlarını belirlenen boyutlara göre eşitliyoruz.

def resize_img(image, size):

    return cv2.resize(image, size)



#calc_Hist fonksiyonu görsellerin okunup renk ve boyut işlemlerini yaptıktan sonra histogramlarını
#hesaplamaktadır.


def calc_Hist(img,size):

    image = cv2.imread(img)
    image = resize_img(image, size)
    image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    image_hist = cv2.calcHist([image_gray], [0], None, [256], [0, 256])

    return image_hist
```

numpy ve opencv kütüphanelerini kullandığımız yollanan resimlerin tek bir boyutta sabitlenmesini sağlayan **resize_img** fonksiyonunu ve yollanan resimlerin histogramını hesaplayan **calc_Hist** fonksiyonunu içermektedir.

compare_images.py Dosyası

```python
from calculate_histogram import *



def compare_imgs(target_img, img_list,size=(100,100)):

    #target image  için histogram hesaplamaktadır

    target_hist = calc_Hist(target_img,size)


    dict_image_hist = dict()

    for i in img_list:

        # karşılaştırılacak görsellerin her biri için historgramları hesaplamaktadır.

        img_hist = calc_Hist(i,size)

       # hedef görsel ile diğer görsellerin histogramlarına göre cosinus benzenzerliğini hesaplanmaktadır.

        cos_sim = np.dot(target_hist.T[0],img_hist.T[0])/(np.linalg.norm(target_hist.T[0])*np.linalg.norm(img_hist.T[0]))

        #hesaplamanın sonuçları key : value   yani path : score olacak şekilde bir sözlük yapısında saklanmaktadır.

        dict_image_hist[i] = cos_sim

    # dict_image_hist adlı sözlükten score değeri en yüksek olan itemi path : score olacak şekilde outuput alıyoruz.


    closest = [k for k, v in dict_image_hist.items() if v == max(dict_image_hist.values())]


    print (closest[0] + " : " + str(dict_image_hist[closest[0]]))
```

calculate_histogram modülünü dahil ettiğimiz target_image , img_list ve size parametrelerini içeren cosinus benzerlik skorunu hesapladığımız **compare_imgs()** fonksiyonunu içermektedir.

Öncelikle hedef resim için histogram hesabı yapılmaktadır.Sonrasında fonksiyona gönderilen resim listesi için for döngüsü kullanılarak her bir resmin histogramı hesaplanmaktadır.

Hedef resmin histogramı ile diğer resimlerin histogramları kosinüs fonksiyonu kullanılarak hesaplanıp sözlük yapısında key : value (görsel path : cosinüs skor) olacak şekilde saklanmaktadır.

Sonrasında ise oluşan sözlükteki en yüksek skora sahip görsel, görsel path : score olacak şekilde output olarak verilmektedir.

result.py Dosyası

```python
import sys

from compare_imgs import *

class CompareImages:


    # args yapısı istediğimiz kadar parametre almak için kullanılmaktadır.

    def __init__(self, args):



        #args[1] ile alınan  birinci parametrenin target image olduğunu belirmektedir.

        self.target_image=args[1]



        # Birinci parametreden sonraki parametreler oluşturulan boş listeye eklemektedir.

        self.other_images=[]

        for i in range(2,len(args)):
            self.other_images.append(args[i])




    def main(self):

        # Oluşturulan parametreler hesaplamaların yapılacağı fonksiyona göndermektedir.

        compare_imgs(target_img=self.target_image, img_list=self.other_images)
```

sys kütüphanesinin ve compare_imgs modülünün kullanıldığı CompareImages sınıfını içermektedir.

def __init__ fonksiyonunda args parametresi kullanılmaktadır.args parametresi fonksiyonun alacağı parametre sayısının belirli olmadığı durumlarda kullanılan bir yapıdır.
Bu fonksiyonda alınan birinci parametrenin (args[1]) hedef görsel olduğu belirtilmektedir.Birinci parametreden sonraki parametreler ise for döngüsü yardımıyla bir listeye eklenmektedir.

“resim1.png” “resim2.png” “resim3.png” bunlar parametreler olarak varsayıldığında “resim1.png”nin hedef görsel ; “resim2.png” ve “resim3.png”nin hedef görsel ile karşılaştırılacak olan görseller olduğu bilinmektedir.

self.target_image=”resim1.png”
self.other_images=[“resim2.png”,resim3.png”]

Bu parametreler de daha sonra main içindeki **compare_imgs** fonksiyonuna parametre olarak gönderilmektedir.

compare_images_run.py Dosyası

```
1   import sys
2   from  result import *
3
4
5   # CompareImages sınıfında nesne türetip sınıfın main fonksiyonunu çağırmaktadır.
6
7   compareImages = CompareImages(sys.argv)
8   compareImages.main()
9
0
```

Son olarak result.py dosyasındaki CompareImages sınıfından bir nesne türeterek main fonksiyonu çalıştırılmaktadır.

terminal tarafından kodun çalıştırılması şu şekilde gerçekleşmektedir.

```
(base) gurayturker@x86_64-apple-darwin13 ImageSimilarity_v0.1 % python compare_images_run.py "/Users/gurayturker/Desktop/Data/Buisness/Travelmean/hotel photos/1030/otelic1.jpg" "/Users/gurayturker/Desktop/Data/Buisness/Travelmean/hotel photos/1030/otelic3.jpg"
/Users/gurayturker/Desktop/Data/Buisness/Travelmean/hotel photos/1030/otelic1.jpg : 0.6682828664779663
```

input ---------> python compare_images_run.py “path1” “path2” “path3”
output ------> path : cosinus_score

path1 hedef resim pathi , diğerleri ise karşılaştırılacak diğer resimlerin pathi olacak şekilde çalıştırılmaktadır.Algoritma 1 hedef 1 karşılaştırılacak görsel olacak şekilde de çalışmaktadır. Karşılaştırılacak görsel sayısı kullanıcının isteğine göre belirlenmektedir.