Phase 1

Group Number: 17

Group Members

Aaryan Gupta

Fenil Madlani

Krisha Gala

Pranav Katariya

Radhika Ganapathy

Shivam Malviya

Abstract

Face Recognition is a computer application that is capable of detecting, tracking, identifying or verifying human faces from an image or video captured using a digital camera. This report introduces the dataset and describes the development stages of face recognition. In this phase, we were given a Olivetti faces dataset. Features such as color moments, extended local binary patterns, and histogram oriented gradients are calculated. Libraries such as numpy, sklearn, skimage were for extracting the features. After extraction of features, comparision of feature vectors was done for each images by using distance functions such as Eucledian distance and Manhattan Distance. This report presents a new method by combing the three feature vectors and getting the most similar image. Our experiment results show that our method is very accurate, reliable and robust for face recognition system.

Keywords : Face Recognition, Color Moments, Extended Local Binary Pattern, Histogram of Oriented Gradients, Skimage, Numpy, Eucledian Distance, Manhattan Distance

Introduction

Humans can easily judge facial attributes through facial information. However, for the computer, it is a very challenging task due to the complex factors such as multiple poses and occlusions in the natural environment.

The Olivetti faces dataset comprises of 400 greyscale face images each of 64x64 pixels. The images are of 40 different person. Each person has 10 different images. The images are captured in different lighting, angles and facial expressions for each subject. In this project, you will experiment with image features, vector models, and similarity/distance.

Our aim is to extract the feature vectors for each image, store them and use them find similarity with the test images. Similarity is calculated based on each feature and combined similarity is also calculated.

Goals

We are given four tasks to perform which are described as follows :-

- Task 1 : Print color moments, elbp, histogram of gradients for an given image ID.
- Task 2 : Given a folder with images, extracts and stores feature descriptors for all the images in the folder.
- Task 3 : Given a folder with images and an image ID, a model, and a value “k”, return most similar k images based on the corresponding visual descriptors.
- Task 4 : Given a folder with images and an image ID and a value “k”, returns and visualizes the most similar k images based on all corresponding visual descriptors.

Proposed Method

First of all I imported all the libraies that I would be using the project. The libraries used in project are :

1. Sklearn
2. Sys
3. Numpy
4. Os
5. Glob
6. PIL
7. Skimage
8. Matplot.lib
9. Scipy.stats
10. Scipy.distance
11. Csv

After importing I created a getImageData function for storing the images into numpy array. Since there were 400 images the shape of numpy array I got was (400,64,64)

1. Proposed Method for Task 1

In this task we need to extract feature descriptors for each image. We start by finding colour moments for the vector.

Color Moment

For extracting the colour moment, I had to split the 64x64 pixel image in smaller block of 8x8 pixels. For this I created an splitting function which takes an array as parameter and performs horizontal and vertical split on it. It returns the split images vector with shape -(400,64,8,8). We calculate the mean, standard deviation and skewness of this vector. All of them have shape of (400,64). We combine mean, standard deviation and skewness into a single vector of dimension -(400,64,3).

Extended Local Binary Pattern

The Elbp vector is calculated using the Local Binary Pattern function. This function is present in skimage library. Input parameters fetched to this function are:

- Image: It is the image vector with shape –(number of images, 64,64) whose elbp vector we are supposed to find
- N : Number of circularly symmetric neighbour set points. I took it as 8 as each center block will have 8 nearby blocks
- R: I have taken radius of circle as 1

- Method : There are four methods to determine the pattern. Since I am using extended lbp, I have taken my method parameter as uniform as it gives improved rotation invariance with uniform patterns which is gray scale and rotational invarient.

The output of the local binary pattern function is an lbp vector of shape –(number of images,64,64). I have stored at in an numpy array.

Histogram Oriented Gradients

HOG is a feature descriptor that is used to detect edges. HOG focuses on the structure of the object. It extracts the information of the edges magnitude as well as the orientation of the edges.
The HOG vector is calculated using the HOG function. This function is present in skimage library. Input parameters fetched to this function are:

- Image : It is the image vector with shape –(number of images, 64,64) whose elbp vector we are supposed to find
- Orientations : 9
- Cell size = 8
- Block size = 2
- Block_norm= 'L2 Hys'. It has an clippling threshold of 0.2

The HOG function gives two outputs – an hog image and a hog vector of shape (num of images,3780).

2. Proposed Method for Task 2

For task 2, we just have to score feature descriptor for each image in an folder. So, I have used csv library to store the results in an csv file. In task 2, I just provide the directory of folder. It then calculates color moments of images in that folder and stores it as vectors. This output file has 5 coloums namely ID, Image name, Color moment vector, Elbp Vector, Hog Vector.

3. Proposed Method for Task 3

In this task we are given we three set of test images. Then nine arguments are given each consisting of set number, image name, model and value of k. Our job is to get the nearest k images for the given image in the given set.
I have started the task by giving an path, k value and a model as input and I fetch the corresponding feature vectors for images in that set. For getting the similarity I have found the distance between two feature vectors for all images with the image given as input.

For Color Moments

Since manhattan distance is well suited to similarity measure based on colour comparision, I have used it to get similar images for this part. Cityblock funtion was used to find the distance.

For ELBP

I have used ecludien distance to compare elbp vectors. I even found distance using KL divergence but I got better results with ecludien distance.

For HOG

For hog, I have again used ecludean distance as it gives the shortest distance.

4. Proposed Method for Task 4

In this task I was asked to find k similar images to a given image using any or all feature descripter. So, for this task, I stored all feature vectores for each image. Then I found the distance between the two images for all features and stored in three different arrays. To normalise these arrays I found out the z score for this array.

$$Z = \frac{x - \mu}{\sigma}$$

Z=Standardised Score
X=observed Value
μ=mean of array
σ =Standard deviation

After finding the zscore all vectors were standardised. So I just assigned weights to each vector and found k similar images while considering all the features.
For assigning weights, I have assigned 0.1 to colour moments feature as there can be images taken in different environment of same person. So I feel color moments should be given less weightage.
Then, I have given weight of 0.45 to Elbp as it measures the texture of image. The texture of image is desireable to a great extent.
For HOG, I have assigned it a weight of 0.45 as it is used to find edges in image. Image edges are desireable in face images as different people have different face, eyes and nose shape. Hence we can distinguish easily with it.

Inference Specification

From the results I inferred that the following conclusions that :-

Comparing based on color moments

It compares the colours in two images and returns the images that have same color combination. Lets say, it two images of different person are taken under similar light conditions it will result as similar image based on colour moments

Comparing based on elbp vectors

It compares the image texture and return best matching image based on the texture. For example, if two people are wearing glasses it will return them as similar to some extent.

Comparing Hog features

It compares the image edges. So it takes into account various facial features such as eyes, eyebrows, nose etc. Different people have different facial features so hog gives excellent results. However if it two different side facing image of same person are compared using hog it will give zero similarity.

This conclusion helped me significantly fetch the similar images from the images given while giving appropriate weights to each feature.

Results

Task 3: Here are few results of task 3 explained

Query 1: set1 image0.png cm8x8 4

Imagename Score

{'Downloads/set1\\image-2.png': 8.687731755352262,

'Downloads/set1\\image-6.png': 10.091387705792938,

'Downloads/set1\\image-7.png': 11.450253446014809,

'Downloads/set1\\image-8.png': 13.406693108722296}

If we compare image 2,6,7,8 are front faces and have similar light conditions hence they turn out to be similar. The lesser the scorer, more is the similarity

Query 2: set1 image0.png elbp 4

Imagename Score

{'Downloads/set1\\image-7.png': 185.1971922033377,

'Downloads/set1\\image-8.png': 194.13139879988503,

'Downloads/set1\\image-2.png': 198.61520586299528,

'Downloads/set1\\image-6.png': 201.02487408278608}

For this set image 7,8,2 and 6 are coming similar as elbp compares texture and all have similar position of their eyes, mouth and other texture.

Query 3: set1 image0.png hog 4

Imagename Score

{'Downloads/set1\\image-7.png': 5.773454462022959,

'Downloads/set1\\image-2.png': 5.9780029295855215,

'Downloads/set1\\image-6.png': 6.128366204423848,

'Downloads/set1\\image-8.png': 7.060448002794174}

For this set image 7, 2,6,8 are coming similar as their image edges and gradient are matching. Hence it comes similar under hog distance

Query 4: set2 image0.png cm8x8 4

Imagename Score

{'Downloads/set2\\image-2.png': 8.687731755352262,

'Downloads/set2\\image-4.png': 17.49842043891185,

'Downloads/set2\\image-1.png': 19.446821360048336,

'Downloads/set2\\image-11.png': 20.5366059690764}

For this set image images 2 ,4 ,1, 11 are coming to be similar as they have macthing color shades. Infact image 1 and 4 are very much alike.

Query 5: set2 image0.png elbp 4

Imagename Score

{'Downloads/set2\\image-2.png': 198.61520586299528,

'Downloads/set2\\image-11.png': 200.06998775428562,

'Downloads/set2\\image-12.png': 200.19990009987518,

'Downloads/set2\\image-4.png': 202.09403751719148}

Query 6: set2 image0.png hog 4

Imagename Score

{'Downloads/set2\\image-2.png': 5.9780029295855215,

'Downloads/set2\\image-1.png': 7.16862703689155,

'Downloads/set2\\image-4.png': 7.319653440339484,

'Downloads/set2\\image-3.png': 7.358685238085616}

The images 2,1,4 and 3 are most similar to the image 0 due to the similar shape and structure of the images from the given set of images.

Query 7: set3 image0.png cm8x8 4

Imagename Score

{'Downloads/set3\\image-70.png': 15.075921991489235,

'Downloads/set3\\image-110.png': 15.15251595392429,

'Downloads/set3\\image-60.png': 16.010005868880587,

'Downloads/set3\\image-90.png': 16.331587769141503}

The given images are similar because they are front facing as well having the same color of the images and the background as well.

Query 8 : set3 image0.png elbp 4

Imagename Score

{'Downloads/set3\\image-90.png': 184.02988887678,

'Downloads/set3\\image-70.png': 184.46408864600178,

'Downloads/set3\\image-80.png': 187.78445090049388,

'Downloads/set3\\image-110.png': 188.8994441495263}

The given images are similar to image 0 as they have the same texture and angle compared to the other images in the given set.

Query 9 : set3 image0.png hog 4

Imagename Score

{'Downloads/set3\\image-110.png': 7.346059152402157,

'Downloads/set3\\image-130.png': 7.452610921785664,

'Downloads/set3\\image-100.png': 7.590557395714776,

'Downloads/set3\\image-120.png': 7.636524391452551}

For this set image 110, 130,100,120 are coming similar as their image edges and gradient are matching. Hence it comes similar under hog distance

Task 4: Here are results of task 4 explained

Query 1: set1 image0.png 4

Image path Z Score

{'Downloads/set1\\image-7.png': -0.10262793985406746,

'Downloads/set1\\image-2.png': -0.01285829959520706,

'Downloads/set1\\image-6.png': 0.061203806924142026,

'Downloads/set1\\image-8.png': 0.28764024021424184}

Set 1 consists of all images of same person and hence the images with front face have least z scores and are most similar

Query 2: set2 image0.png 4

Image path Z Score

{'Downloads/set2\\image-2.png': -0.17433190760751588,

'Downloads/set2\\image-4.png': 0.27184965268861633,

'Downloads/set2\\image-1.png': 0.3422484015829996,

'Downloads/set2\\image-3.png': 0.34574513045679867}

Set 2 consists of images of two different person and hence the images of same person comes at output.

Query 3: set3 image0.png 4

Image path Z Score

{'Downloads/set3\\image-110.png': 0.04908222336957168,

'Downloads/set3\\image-70.png': 0.16299589296356848,

'Downloads/set3\\image-90.png': 0.2186113303888444,

'Downloads/set3\\image-100.png': 0.25447183670644086}

Set 3 consists of images of all different person and hence the images of person who look similar comes at output.

Bibliography

https://scikit-learn.org/0.19/datasets/olivetti faces.html

https://en.wikipedia.org/wiki/Color_moments

http://researchspace.csir.co.za/dspace/handle/10204/6491

https://en.wikipedia.org/wiki/Local_binary_patterns

https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients