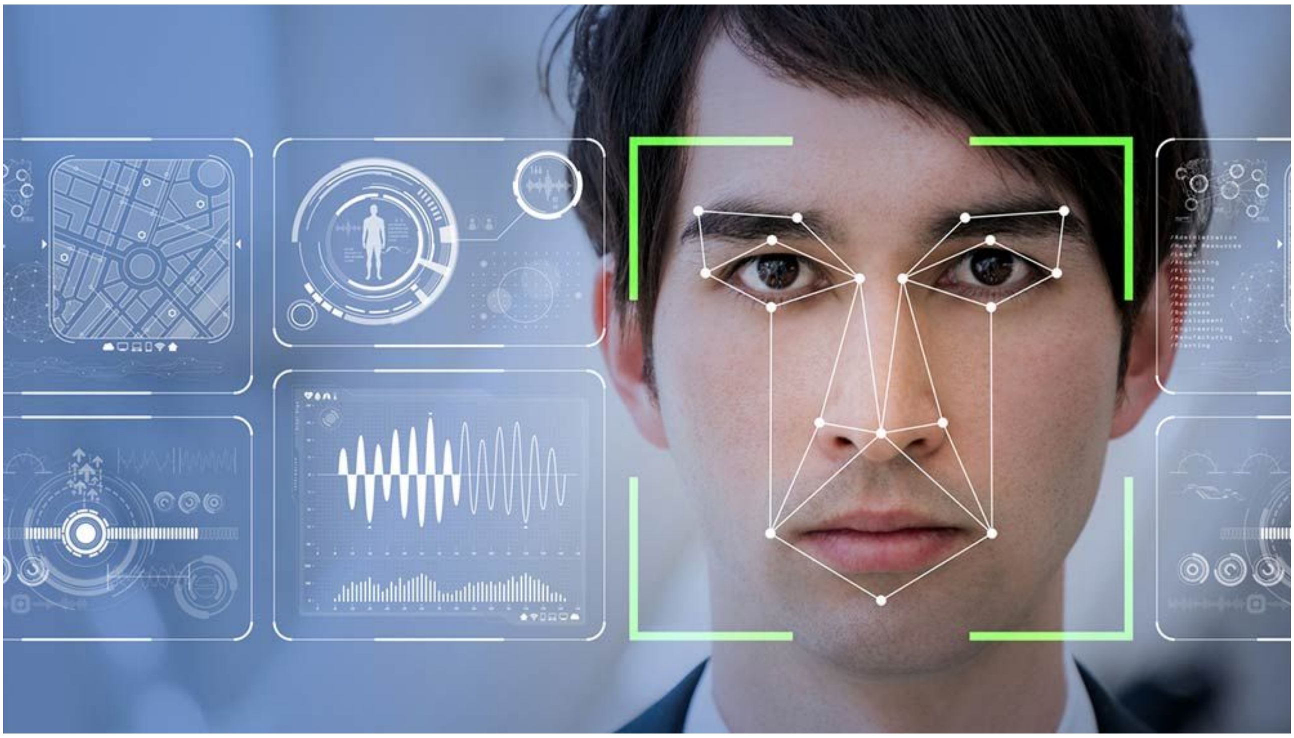
Face Recognition

Using OpenCV

Sohail Khan (B16CS036) Harshul Sharma (B16ME011)
Mentor: Dr. Chiranjoy Chattopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Jodhpur



Contents:

Introduction:

A facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a videosource.

Face recognition is still a very demanding area of research. This problem becomes more challenging in unconstrained environment and in the presence of several variations like pose, illumination, expression, etc.

The face recognition approaches are broadly classified into two major categories,
Deep learning based face recognition and descriptor based face recognition. Some recent deep learning based approaches are FaceNet and DeepFace for face recognition. The deep learning based approaches are being popular due to high performance but at the cost of increased complexity of training in terms of the time, computing power and data size. The deep learning based approaches are also biased towards the training data.

Motivation:

The motivation behind this project is that facial recognition has an amplitude of possible applications. Webcams are often used as a security measure for locking a personal computer. The webcam's facial recognition technology allows for the computer to be accessible to the user only if it recognizes their face. Facial recognition technology can also be used to keep track of the attendance of the students.
Facial recognition is the identification of humans by the unique characteristics of their faces. Facial recognition technology is the least intrusive and fastest biometric technology.It works with the most obvious individual identifier the human face. With increasing security needs and with advancement in technology extracting information has become much simpler. This project aims on building an application based on face recognition using algorithms and comparing the results with different databases.

The descriptor based face recognition approaches can be divided into learning based descriptors and handcrafted descriptors.The hand-designed local descriptors are very simple from design aspect. This class of descriptors have shown very promising performance in most of the computer vision problems.

The main advantages of the handcrafted local descriptors are as follows:
a) it is not dependent upon the database,
b) it does not require very complex computing facility, and
c) lower dimensional descriptors can boost the time efficiency significantly.

Objective:

1. The basic purpose being to identify the face and retrieving information stored in database. It involves two main steps. First to identify the distinguishing factors in image and storing them and Second step to compare it with the existing images and returning the data related to that image

2. Trying to find a face within a large database of faces. In this approach the system returns a possible list of faces from the database. The most useful applications contain crowd surveillance, video content indexing, personal identification (example: drivers license), mugshots matching, etc.

3. Real time face recognition: Here, face recognition is used to identify a person on the spot and grant access to a building or a compound, thus avoiding security hassles. In this case the face is compared against a multiple training samples of a person

Challenges/Research Issues:

The problem of Facial recognition becomes more challenging in unconstrained (wild) environment and in the presence of several variations like :

1. Illumination variations
2. Pose variations
3. Facial expressions
4. Facial hair changes
5. Age changes
6. Face wearable changes

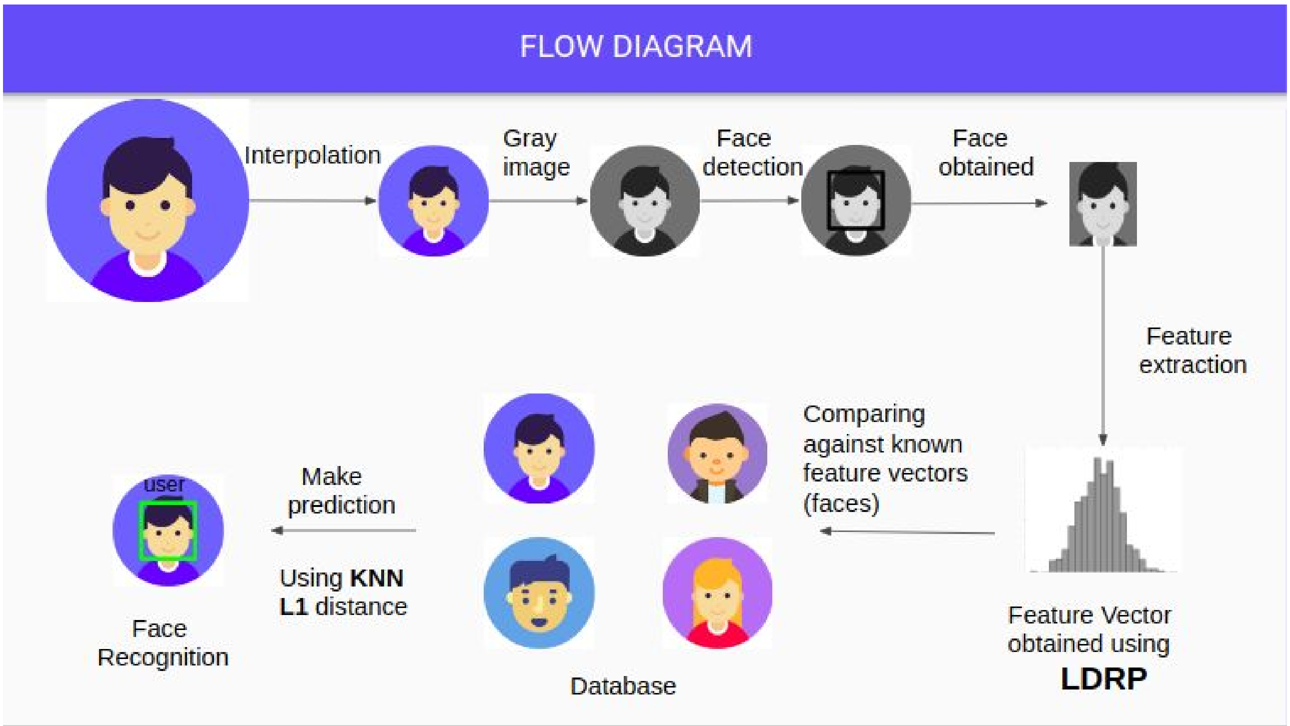
Basic overview of methodology:

The Facial recognition problem is basically divided into 4 major steps:

1. **Find a face in a image(Face detection):** Face detection in this project is done by using openCVs built in pre-trained facial detector, namely, haar_cascade_classifier and LBP_cascade_classifier.

2. **Extract facial features from the face image:** The project uses descriptor based methodology. Feature vector from the face image is computed using LDRP (Local Directional Relation Pattern) which encodes the relationship among directional neighbors and then utilizes the encoded values with central pixel to generate final pattern.
3. **Compare against known faces:** The feature vector is computed for the input(testing) face image, and then the obtained feature vector is compared with the existing list of feature vectors of the trained images by KNN (kth nearest neighbor) using Eucledien distance.

4. **Make a prediction (Recognition):** Based on the distances computed with the list of feature vector, K nearest neighbors of the input feature vector are obtained. Then the prediction is made based on the votes given to the K nearest neighbors.



Face Detection:

Face detection involves detecting the face area from the input image. Face detection process is used as for facial recognition, only the face regions is important, and this is what imparts distinctive features to individuals.

Face detection in this project is done using openCV's built in face detection classifiers.
A classifier is basically a computer program which decides whether an image is a positive (face) image or negative(non-face). The classifier is trained on a large dataset of face images before it can be used for successful detections.
OpenCV provides few pre-trained face classifiers, which are as follows:

1.) Haar classifier

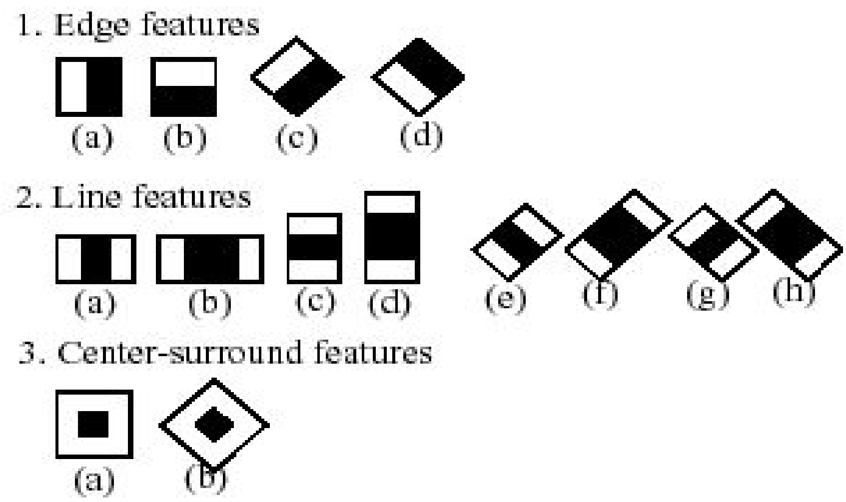
2.) LBP classifier

The classifiers process image in grayscale, as we don’t need color information to determine whether an image is a face image or not.
As they are pre-trained we must also include their knowledge files first.
openCV provides a set of knowledge files for various applications.
eg:- *haarcascade_frontalface_alt.xml. , haarcascade_frontalface_alt2.xml., LBPcascade_frontalface_alt.xml.*

Haar feature based cascade classifier:

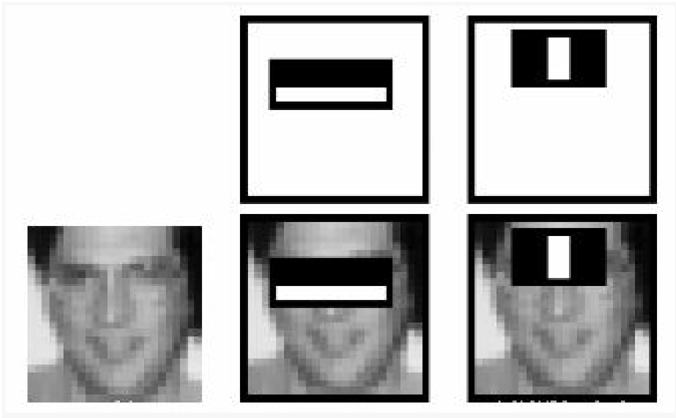
It is a ML based approach, which was proposed by Paul Viola and Michael Jones.
It extracts HAAR features from the input image.

The haar features are as follows:



Each window is placed on the picture to calculate a single feature. This feature is a single value obtained by subtracting the sum of pixels under the white part of the window from the sum of the pixels under the black part of the window.

Now, all possible sizes of each window are placed on all possible locations of each image to calculate plenty of features.

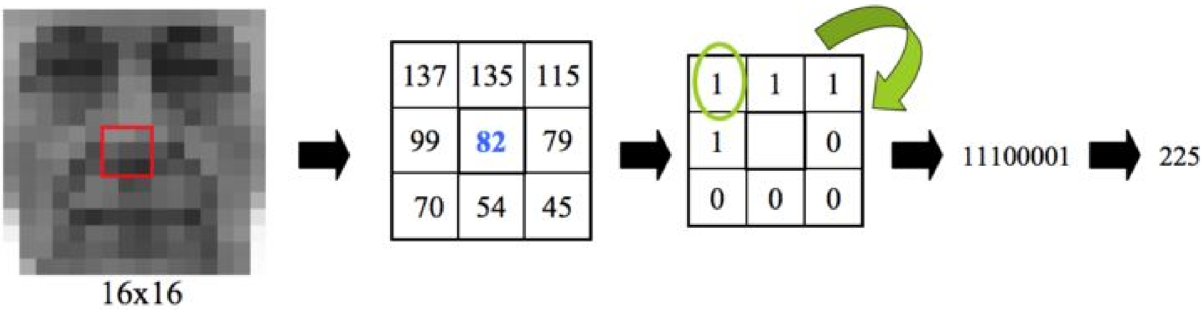


LBP cascade classifier:

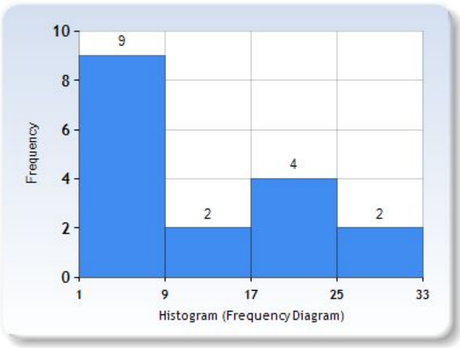
This classifier uses Local Binary pattern face descriptor algorithm.

LBP features are extracted to form a feature vector that classifies a face from a non-face. In LBP Cascade, a 3X3 pixel window is made to traverse the whole image, with the central pixel being utmost important for each window.
Then, the central pixel value is compared with every neighbor's pixel value under the 3×3 window. For each neighbor pixel that is greater than or equal to the center pixel, LBP sets its value to 1, and for the others, it sets them to 0.

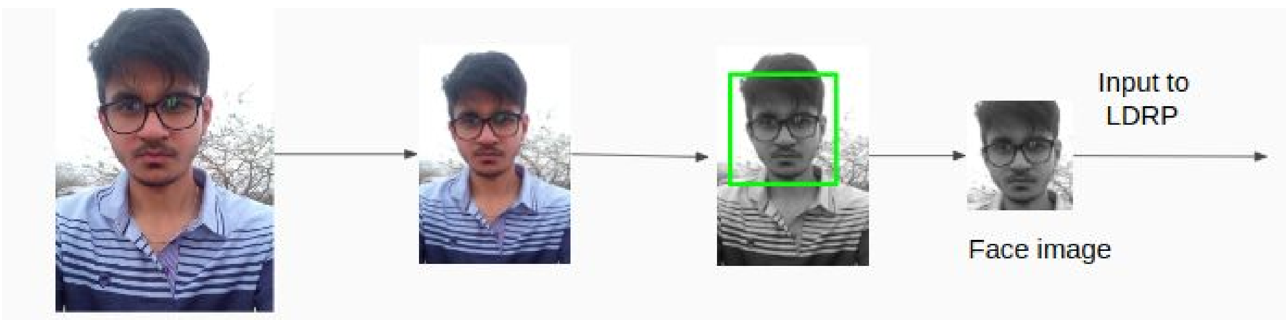
After that, it reads the updated pixel values (which can be either 0 or 1) in a clockwise order and forms a binary number. Next, it converts the binary number into a decimal number, and that decimal number is the new value of the center pixel. We do this for every pixel in the image.



Now, after you get a list of local binary patterns, you convert each one into a decimal number using binary to decimal conversion (as shown in above image) and then you make a histogram of all of those decimal values. A sample histogram looks like this:



In the end, you will have one histogram for each face in the training data set. That means that if there were 100 images in the training data set then LBPH will extract 100 histograms after training and store them



Recognition:

For facial recognition process feature vectors are extracted from the image.

LDRP Descriptor:

The construction process of local directional relation pattern is divided into following steps

Local Neighborhood Extraction:

Let, I is an image with dimension x × y and I i,j represents the intensity value for the pixel in ith row and jth column with **i ∈ [1, x]** and **j ∈ [1, y].** The coordinates of top and left corner is (0, 0) with positive x-axis downside across the rows and positive y-axis right side across the columns. The N local r neighbors of I (i,j) at a radius r are represented by **I** (i,j), where r, kth I (i,j) is the k neighbor with **k ∈ [1, N]**.The coordinates of k th neighbor of pixel (i, j) at a radius r is given by (i (k) , j (k)) defined as follows

$$i_k = i + r\cos\theta_k$$
$$j_k = j - r\sin\theta_k$$

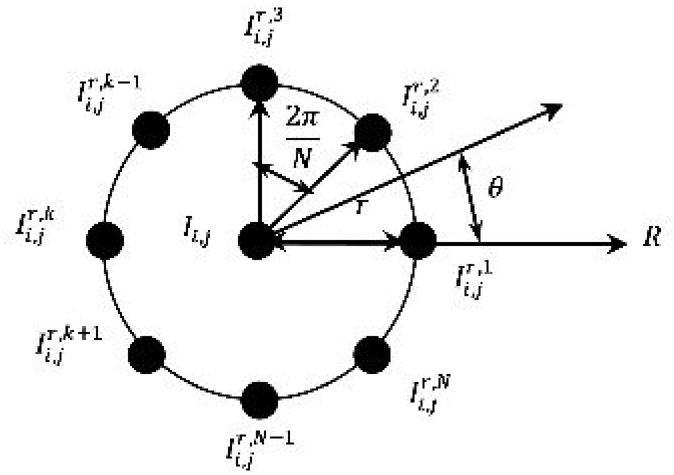
where **θ (k)** is the angular displacement of k th neighbor w.r.t. first neighbor and given as follows

$$\theta_k = (k-1) \times \frac{360}{N}$$

So, I (i,j) can be written as follows,

$$I_{i,j}^{r,k} = I_{i_k,j_k}$$

The first neighbor is considered in the right side of the
center pixel and rest of the neighbors are computed w.r.t. first
neighbor in the counterclockwise direction



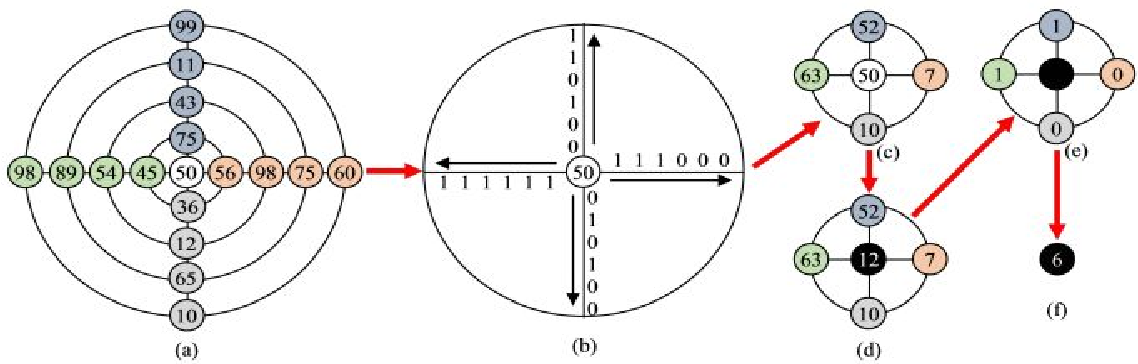
Local Directional Information Coding :

In order to increase the discriminative ability of the proposed descriptor, the wider neighborhood is used in this work. The relation among local neighbors at multiple radius are utilized to encode the directional information. The θ (k) represents the k th direction with **k ∈ [1, N].** Considering the M directional neighbors in k th direction, the binary codes are computed between each pair. The number of pairs out of Mth neighbors are (M c 2). Let, the t th directional neighboring pair in k th direction is represented by (I (r 1 ,k,i,j) , I (r 2 ,k,i,j). The index for t th pair can be computed from r 1 and r 2 as follows

$$t = \begin{cases} r_2 - r_1, & \text{if } r_1 = 1; \\ r_2 - r_1 + \sum_{\eta=1}^{r_1-1}(m-\eta), & \text{otherwise.} \end{cases}$$

where **t ∈ [1, (M c 2)], r 1 ∈ [1, M − 1]** and **r 2 ∈ [r (1) + 1, M]**.Let, **β (k,i,j)** denotes the local directional binary pattern for center pixel (i, j) in k th direction. The binary code between t th directional neighboring pair (or between the two neighbors at radius r 1 and r2) in k th direction for center pixel (i, j) is generated as follows

$$\beta_{i,j}^{k}(t) = \begin{cases} 1, & \text{if } I_{i,j}^{r_1,k} \leq I_{i,j}^{r_2,k}; \\ 0, & \text{otherwise.} \end{cases}$$



For M neighbors in a direction μ = (M c 2) number of binary values are generated. In order to reduce the dimension of the descriptor, it is required to code these binary values into a single value. A local directional information code (Γ) is generated in each direction from the binary values in that direction. The local directional information code, Γ, in k th direction for pixel (i, j) is computed by the following equation,

$$\Gamma_{i,j}^{k} = \sum_{\eta=1}^{\mu} (\beta_{i,j}^{k}(\eta) \times \xi(\eta))$$

where, ξ is a weight function and defined as follows

$$\xi(\eta) = 2^{\eta-1}.$$

Local Directional Relational Pattern :

The local directional relation code is computed in the previous subsection for a direction by encoding the relationship among the neighbors at different radius in that direction. Now, the next step is to find out the relation between center pixel and local directional relation codes. The minimum and maximum values of local directional code are dependent upon the number of directional neighbors considered (i.e. M). The code is generated from the μ = (M c 2) number of binary values. So, the different number of decimal values that can be generated from μ binary bits is 2 μ with a minimum value as 0 and maximum value as 2 μ − 1. Whereas, the minimum and maximum values of center pixel are 0 and 2 B − 1 respectively, where B is the bit-depth of the image. Note that, the bit-depth (B) of the images is 8 in the databases used in this paper. A clear mismatch can be observed between the range of center pixel and local directional relation code. Thus, a transformation is required over either the center pixel or the local directional relation codes to match both the ranges. Due to efficiency reason, the center pixel is transformed into the range of local directional relation codes as follows

$$\tau_{i,j} = \Upsilon(I_{i,j} \times \frac{2^{\mu}-1}{2^{B}-1}).$$

where, τ (i,j) is the transformed version of I (i,j) , Y(η) is a function to round η to the closest integer value. The transformed value of center pixel for μ = 6 and B = 8 is
6 computed as.

$$\tau_{i,j} = \Upsilon(50 \times \tfrac{2^{6}-1}{2^{8}-1}) = 12.$$

Let, ρ is a binary pattern representing the relationship between center and directional relation code having N values corresponding to each direction. The ρ (N,M i,j) (k) for center pixel (i, j) in k th direction is given as follows

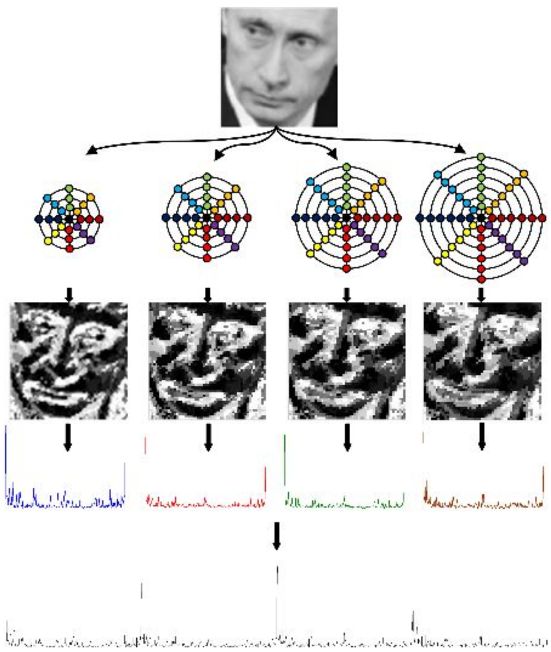
$$\rho_{i,j}^{N,M}(k) = \begin{cases} 1, & \text{if } \Delta_{i,j}^{k} \geq 0; \\ 0, & \text{otherwise.} \end{cases}$$

where, ∆ (k,i,j) is the difference between local directional relation code in k th direction and transformed value of the center pixel,

$$\Delta_{i,j}^{k} = \Gamma_{i,j}^{k} - \tau_{i,j}$$

The local directional relation pattern (LDRP) for pixel (i, j) by considering the local neighbors in N directions with M neighbors in each direction is computed as follows

$$LDRP_{i,j}^{N,M} = \sum_{k=1}^{N} (\rho_{i,j}^{N,M}(k)) \times \xi(k)),$$



LDRP Feature Vector :

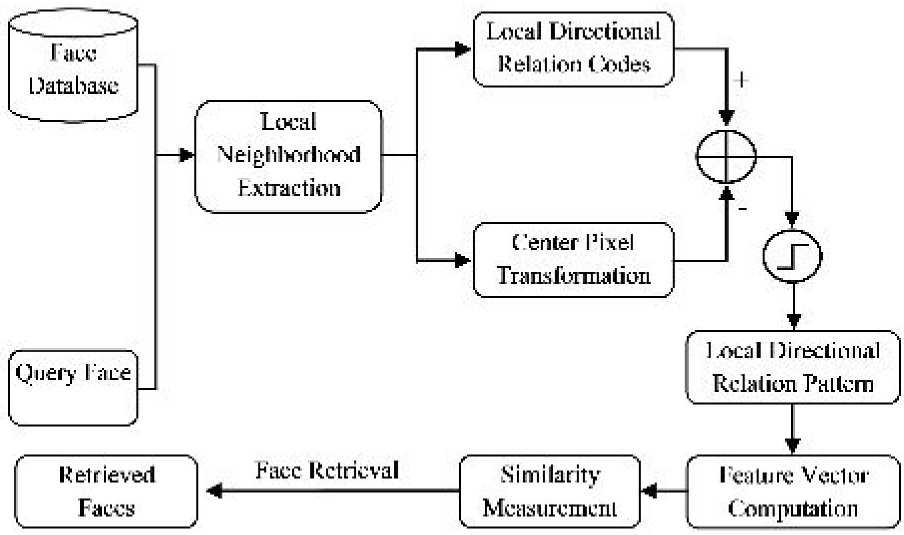
The LDRP feature vector (H) is generated by finding the number of occurrences of LDRP values over the whole image. Note that, the minimum and maximum values of LDRP are 0 and 2 N − 1 respectively. Thus, the length of feature vector is 2 N . The LDRP feature vector for Image I with local neighborhood from N directions having M neighbors in each direction is defined as follows,

$$H^{N,M}(\eta) = \sum_{i=M+1}^{x-M} \sum_{j=M+1}^{y-M} \zeta(LDRP_{i,j}^{N,M}, \eta)$$

where, ζ is calculated by following rule

$$\zeta(\alpha_1, \alpha_2) = \begin{cases} 1, & \text{if } \alpha_1 = \alpha_2; \\ 0, & \text{otherwise.} \end{cases}$$

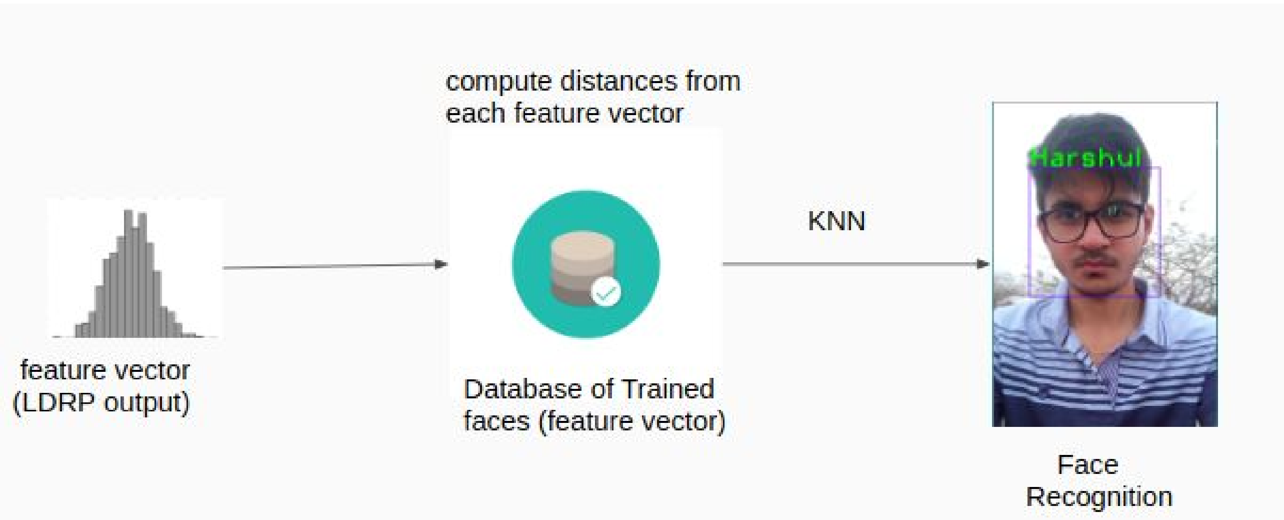
Multi-scale LDRP:



In order to make the final feature vector invariant to the image resolution, H (N,M 1 ,M 2) is normalized as follows

$$H^{N,M_1,M_2}_{Normalized}(\delta) = \frac{H^{N,M_1,M_2}(\delta)}{\sum_{\lambda=1}^{d^{N,M_1,M_2}} H^{N,M_1,M_2}(\lambda)}$$

∀δ ∈ [1, d N,M 1 ,M 2]. In the experiments, the normalized version of feature vector is considered for all descriptors. An image is considered from the LFW database [58]. Four LDRP maps are created by considering M = 3, 4, 5, and 6 number of directional neighbors (i.e., M 1 = 3 and M 2 = 6) in N = 8 directions. The final LDRP feature vector is computed by concatenating the feature vectors of different LDRP maps. In the rest of this paper, the by-default values of parameters of LDRP descriptor are B = 8, N = 8, M 1 = 3, and M 2 = 6.



Prediction:

The prediction is made based on the votes given to the K nearest neighbors (KNN).In pattern recognition, the *k*-nearest neighbors algorithm (*k*-NN) is a non-parametric method used for classification. In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.

The model for kNN is the entire training dataset. When a prediction is required for a unseen data instance, the kNN algorithm will search through the training dataset for the k-most similar instances. The prediction attribute of the most similar instances is summarized and returned as the prediction for the unseen instance.

The similarity measure is dependent on the type of data. For real-valued data, the Euclidean distance can be used. Other other types of data such as categorical or binary data, Hamming distance can be used.

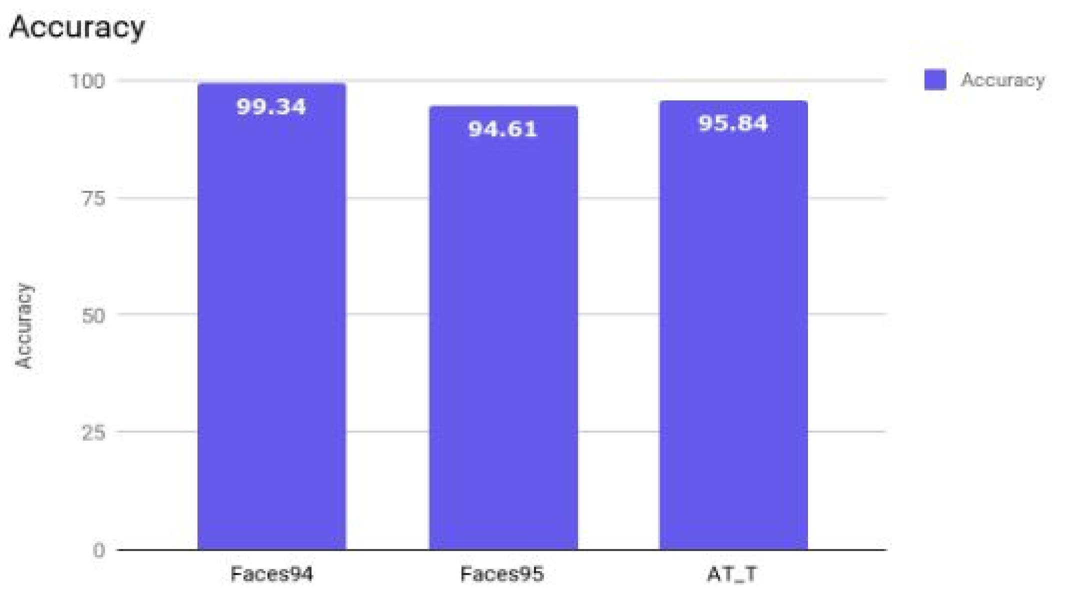
In the case of regression problems, the average of the predicted attribute may be returned. In the case of classification, the most prevalent class may be returned.

Results:

The descriptor is evaluated under image retrieval framework for various face databases like ESSEX - Face94 and Face95 , AT&T, etc. The face94 had 50 subjects and each subject had 20 images with expression variation. The face95 had 30 subjects and each subject had 20 images with expressions, light variations, and slight pose variations. AT&T had 40 subjects with 10 images for each subjects with expressions, light variations, and slight pose variations (with and without glasses).

The descriptor gave promising results for all the above mentioned databases and recognised faces under unconstrained environments.

Datasets	Training Images	Testing Images	Correct Matches
Faces94	848	152	151
Faces95	450	130	123
AT&T	280	120	115



Comparison with existing Descriptors:

- Existing descriptors: **LBP, LDP, LDGP, LGHP**.
- Feature vector dimensions = 256, 1024, 512, 9216, respectively.
- Dimension of **LDRP** feature vector = 1024, and is independent of size of image, unlike **HOG**.
- LGHP also works on similar principle, but the maths included and the size of feature vector decreases the time and space efficiency significantly.

Conclusion:

LDRP is a robust facial feature retrieval descriptor which can be used to recognise faces under unconstrained environment with pose, light , expression variations. LDRP is also invariant of image size and computes a feature vector of dimension 1X1024 which is a good number as compared to the other existing facial feature descriptor.

References:

[1] Soumendu Chakraborty; Satish Kumar Singh; Pavan Chakraborty, Local Gradient Hexa Pattern: A Descriptor for Face Recognition and Retrieval, IEEE Transactions on Circuits and Systems for Video Technology, Volume: 28, Issue: 1, Jan. 2018.

[2] Shiv Ram Dubey, Local Directional Relation Pattern for Unconstrained and Robust Face Retrieval, arXiv:1709.09518, 2017.

[3] HOG: https://www.learnopencv.com/histogram-of-oriented-gradients/

[4] LBP: https://ieeexplore.ieee.org/document/7322425/

[5] Wikipedia : https://en.wikipedia.org/

[6] Haar Cascade, OpenCV: https://docs.opencv.org/2.4/index.html

[7] Blog: https://www.superdatascience.com/opencv-face-recognition/