Face Detection and Recognition

1st Parag Poddar
2019BCS-038
paragpoddar123@gmail.com
IIITM
Gwalior, India

2nd Shivaji Kumar
2019BCS-058
official.shivaji007@gmail.com
IIITM
Gwalior, India

3rd Sanket Kumar
2019BCS-054
sanketdawar5875@gmail.com
IIITM
Gwalior, India

***Abstract*—Face detection and recognition is bio-metric technology, which involves the identification of the faces of human and recognize the person. The paper contains the related researches and studies with different perspective of face recognition. The paper unfolds the development stages and all the other related technologies based on face recognition. We wanted to give forward looking view to face recognition. There are already a lot of research and work done on face recognition. Face recognition has become the future development direction and has many potential application prospects.**

***Index Terms*—Haar Cascade, training, testing, classifier, features**

I. Introduction

In the current time, as the technology is getting better and better, images or live camera has became a most convenient and preferable mode of expressions. Various photos are uploaded everyday in different clouds and social media networks. A very challenging task is to retrieve and organize these photos which also impact user experience.

For this problem, there are some solutions like geo-tagging, that allows to organise the photos by locations but it becomes very difficult while doing so with some queries like ”photos with some particular friends”. The reason is that this kind of queries require human face detection and recognition.

Photographs or live webcam are the 2D projections of 3D objects like faces, any other object etc. There are billions of faces and each and every face has something different in it. Inter-personal variations in human faces can be due to race, colour, geographical locations, genetics, identity etc whereas intra-personal variations in human faces can be due to their expressions, hair style, angle of the face, etc. We have taken Face recognition problem for this course project. Recognising any individual person by his or her is an easy task for humans but its not the case with technologies. Its a challenge for vission-based automated system.

Face detection and recognition has been a topic of active research for a long period of time. There are well defined and developed technologies are available for human face detection and recognition as well in a photo or real time using web camera. A very easy example of it is the face recognition system in every smart phone that recognise before getting unlocked that if the user is the actual owner of that smartphone. However, a demerit of these technologies is that it fails to detect faces in different angles or partial faces.

In this project, we have tried to develop Face Detection project. In this project, we are recognising the faces real time using webcam with the help of Haar Cascade Model. Haar Cascade Model for face detection is very well known to everyone. We have tried to train the model and detect the faces.

II. Literary Survey

A. Facial Recognition and its origins

Facial recognition is a method of identification or verification of a person by their face in a computer application usually using machine learning. Face recognition systems uses distinctive details/features such as eyes, eyebrows, nose, chin, color of skin, etc. of a person’s face to identify a face. These features are then turned into mathematical data and compared with the facial database.

Facial recognition system was first developed in 1960’s by Computer scientist and mathematician Woodorn Wilson Blendsoe. Because of his work he is also known as unofficial father of facial recognition technology. Not much of his work was published in the public because the funding was provided by an unnamed intelligence agency. Peter Hart continued the work of Bledsoe in Stanford Research Institute and performed an experiment in 1996, where computer outperformed humans consistently [1]. But as computers grew powerful in 2010’s facial recognition systems became more and more efficient.

Present Facial recognition is a three step process: 1: Face Detection 2: Feature Extraction 3: Face Recognition [2]. Lets now discuss them in detail.

B. Face Detection and Feature Extraction

There are many machine learning models present in form of libraries where face detection and feature extraction can be employed easily. Some of the models we tested are Dlib, MTCNN, Haar Cascade.

1) Dlib: Dlib is a toolkit in C++ which is used for making real world machine learning and data analysis applications, it’s written in C++ but also has python bindings which can be easily used in python. The face detector in this method is based on Histogram of Oriented Gradient(HOG) and linear SVM. Dlib is only good for ‘frontal face’ as odd angles are not easily detected. The frontal face detector works in the following manner. First

features are extracted by HOG then these features are passed through SVM. In HOG, distribution of the gradient is used as features.

Dlib also provides CNN(conventional Neural Network) based face detector which is capable of detecting faces almost in all the angles, but it does not work in real-time on the CPU. In Dlib, the implementation of the algorithms are totally separated from the data on which they operate. This makes dlib generic enough to operate on any kind of data. [3]

2) MTCNN: MTCNN was introduced as a python library by Kaipeng Zhang in paper "Joint Face Detection and Alignment Using Multi-task Cascading Convolutional Networks" [4]. It leverages a 3- stage neural network detector.

First, convolutional network is used to obtain candidate windows along with their boundary box regression vectors, and then highly overlapped candidate windows are overlapped using non-maximum suppression(NMS). Then these candidate windows are passed through other CNN which rejects large false positives and performs calibration of bounding boxes. Then in the final stage, the facial landmark detection is performed. The advantage of MTCNN over other methods is that where other methods can only detect faces of a particular size in image, MTCNN tries to detect the images of different sizes. [4]

3) Haar Cascade: Haar cascade is a feature-based cascade classifier.It is a machine learning based approach where a cascade function is trained using lots of positive and negative images, which are then used to detect other images.

First it extracts lots of features in the images, then best features are selected. This reduces 160000 features into 6000 features. Then a concept called 'cascade of classifiers' is used where instead of using thousands of features in a window at the same time, features are grouped into different stages of classifiers and applied one-by-one in different stages. This makes Haar cascade faster than other methods. There are

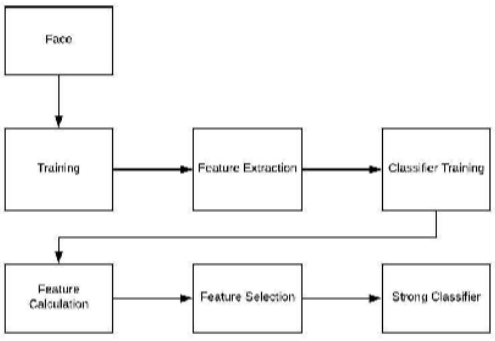

Fig. 1. Haar Cascade Flow Chart [5]

2 image databases used in the Haar Model. The FEI face database is a Brazilian database containing 14 images for each of 200 individuals, with a total of 2800 images. The yale face database contains facial images of 15 individuals, with 11 pictures per person, taken with different illumination conditions. These classifiers use haar-like features that are applied over the image. Only those image regions, called sub-windows, that pass through all the stages of the detector are considered to contain the target object [6]. In this project we are using Haar Cascade due to its ease of use and computation efficiency.

III. Methodology

A. System used

So as discussed above the facial recognition system works in three steps face detection, feature extraction and face recognition. Face detection and feature extraction are done simultaneously [2]. As we are using live video capture for the

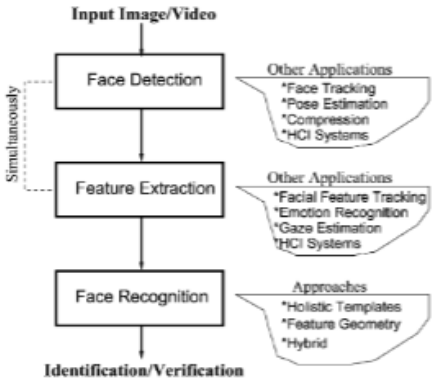

Fig. 2. Configuration of generic face recognition system

users face detection, we follow the system discussed in the paper 'Image-based Face Detection and Recognition "State of the art" [7]. In this system we first create a data set of the face images then detect the face using a machine learning model, then we frame and extract the face/ facial data, then we apply some processing on the data. Once these steps are done we train the data set and create a trained classifier. Once these steps are completed, our facial recognition system is ready to use and now we can test it by querying an image to the dataset which goes through trained classifier and we get the accuracy/confidence of the system.

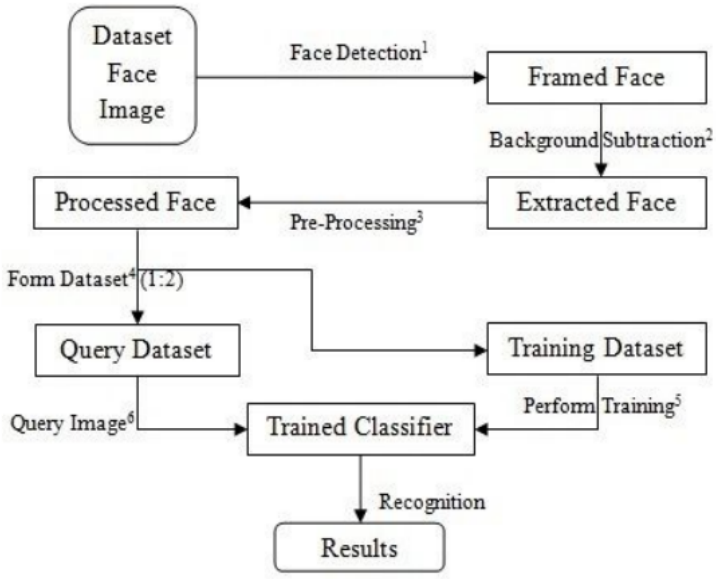

Fig. 3. System used for facial recognition

B. Walkthrough of the project

1) Face detection: In first step we use OpenCV to access the camera and take the pictures of the face to be detected. Then we use haar cascade to identify the face in the image. once a face is detected, the image is captured and is then saved into the distinct dataset for each user.

```
#start detect your face and take 50 pictures
while(True):

    ret, img = cam.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_detector.detectMultiScale(gray, 1.3, 5)

    for (x,y,w,h) in faces:

     \cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0), 2)
     count += 1

     # Save the captured image into the datasets folder
     cv2.imwrite(path+'/'+str(count)+".jpg",gray[y:y+h,x:x+w])

     cv2.imshow('image', img)

    k = cv2.waitKey(100) & 0xff
    # Press 'ESC' for exiting video
    if k == 27:
        break
    elif count >= 50: # Take 50 face sample and stop video
         break
```

2) Feature extraction/Training: Our program goes through each image present in dataset, then haar cascade method generates a numpy array and we also extract label using the file name. Then we pass our numpy array as training feature and label as training label. After iterating to all the images in dataset we dump label array in file called pickle. And our trained model is saved as trainer.yml file in trainer folder.

```
for root, dirs, files in os.walk(image_dir):
for file in files:

if file.endswith("jpg"):
path = os.path.join(root, file)
label = os.path.basename(root).replace(" ", "-").lower()
# print(label, path)
if not label in label_ids:
label_ids[label] = current_id
current_id += 1
id_ = label_ids[label]
#print(label_ids)
#y_labels.append(label) # some number
#x_train.append(path)
# verify this image, turn into a NUMPY arrray, GRAY
pil_image = Image.open(path).convert("L") # grayscale
size = (550, 550)
final_image = pil_image.resize(size, Image.ANTIALIAS)
image_array = np.array(final_image, "uint8")
#print(image_array)
faces = face_cascade.detectMultiScale(image_array,
scaleFactor=1.5, minNeighbors=5)

for (x,y,w,h) in faces:
roi = image_array[y:y+h, x:x+w]
x_train.append(roi)
y_labels.append(id_)

with open("pickles/face-labels.pickle", 'wb') as f:
pickle.dump(label_ids, f)

recognizer.train(x_train, np.array(y_labels))
recognizer.write("trainer/trainer.yml")
```

3) Face recognition: In last step we use the camera to capture the video and use each frame as an image and extract available face using haar cascade method as a numpy array. Now this numpy array is provided to recogniser model which we have generated in previous step. Then recogniser provides us with an confidence and id of label. With this id we can find the user using the array stored in pickle file.

```
while True:
 ret, img =cam.read()
 gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
 faces = faceCascade.detectMultiScale(
  gray,
  scaleFactor = 1.2,
  minNeighbors = 5,
  minSize = (int(minW), int(minH)),
  )
for(x,y,w,h) in faces:
 cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)
 id, confidence = recognizer.predict(gray[y:y+h,x:x+w])
 # Check if confidence is less then 100 ==> "0" is perfect
 if (confidence < 100 and confidence>0):
     id = labels[id]
     confidence = "  {0}%".format(round(100 - confidence))
 else:
     id = "unknown"
     confidence = "  {0}%".format(round(100 - confidence))

 cv2.putText(img, str(id),(x+5,y-5),font,1,(0,0,255),2)
 cv2.putText(img,str(confidence),
 (x+5,y+h-5),font,1,(0,0,255),1)

cv2.imshow('camera',img)
k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video
if k == 27:
    break
```

IV. Experiment and Results

A. Dataset

In this project, we have made our own dataset. Usually the datasets are downloaded from different sources but here we decided to make our own dataset because the face is being recognised in a webcam. The Dataset is created with different amount of photos from each person for better results. The amount of photos we want in our dataset for training can be

decided by ourself. It can be simply changed by editing in the code. Starting from 30, we performed the experiment till 500 photos per person in the dataset. The results for each dataset can be seen below.

B. Results

After the development stages, the model was tested with different faces and different amount of faces in dataset. We started with very less amount of photos for each person in the dataset, i.e, 30, but that definitely lacked in the accuracy. Then we increased the amount to 50. That gave a comparatively better results than before but it was also not enough. Then we decided to test the model for 100 photos for each person in the dataset with different expressions and angles. Similarly, for better results, we increased the count to 300 and then 500.

No. of samples	Observation-1	Observation-2	Observation-3	Average %
30	24%	26%	22%	24%
50	26%	24%	27%	25.6%
100	34%	36%	33%	34.3%
300	39%	39%	40%	39.3%
500	42%	41%	41%	41.3%

Fig. 4. Accuracy for different amount of sample photos in dataset

The table shows that we have calculated accuracy three times with same amount of photos in dataset for a particular person. The average accuracy with only 30 photos was 24 percentages, which is comparatively very low. Then, with 50 photos, the accuracy increases to 25.6 percentages, which is better then previous but still not satisfactory. Then, with 100 photos, the accuracy increases to 34.3 percentages, a drastic change in the accuracy due to double no. of photos than previous. Then with 300 photos, we again have increment in the accuracy to 39.3 percentages. Then with 500 photos, we got accuracy of 41.3 percentages. Since, we were getting very low accuracies, we decided to increase the input samples to a large amount. Therefore, we first increased it to 5000 images per person. That gave us the accuracy of 69 percentages. Then atlast we tried with a sample of 10000 images and that gave us accuracy of 75 percentages on an average. No doubt, the final accuracy is low, but the accuracy has improved a lot from where we started.

C. Individual Contributions

Contributions from each group member are almost equal. All the group members were reading some research papers in the first half of the project and then we went through the strong points of each member. Sanket kumar has done the face detection part. Shivaji kumar has done the training and recognition coding part with the discussion among all group members. Parag Poddar has done experimentation,testing, and debugging of the code. After the coding part, the major section of the report is written by Sanket Kumar and Parag Poddar with some contribution from Shivaji Kumar.

D. Conclusion

The Face recognition is a supervised learning problem in machine learning. Here the faces with corresponding features can be treated as samples input and corresponding name of the person is label. Through this project, we executed the face recognition using Haar cascade model. The model was trained using dataset with different amount of sample photos and then tested for the same person. The accuracy with lower amount of photos in dataset is pretty bad but as the amount of sample photos increases the accuracy increases.

ACKNOWLEDGMENT

We would like to thank our professor Dr. Sunil Kumar, for this opportunity to deeply understand and work on this topic. Again, we are very thankful to him for his guidance throughout the project.

REFERENCES

[1] S.B.Thorat, S. Nayak, and J. Dandale, "Facial recognition technology: An analysis with scope in india," *International Journal of Computer Science and Information Security*, vol. 8, 04 2010.

[2] W.-Y. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, pp. 399–458, 12 2003.

[3] D. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 07 2009.

[4] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, 04 2016.

[5] R. Hasan and A. Sallow, "Face detection and recognition using opencv," *Journal of Soft Computing and Data Mining*, vol. 2, 10 2021.

[6] R. Padilla, C. Filho, and M. Costa, "Evaluation of haar cascade classifiers for face detection," 04 2012.

[7] F. Ahmad, A. Najam, and Z. Ahmed, "Image-based face detection and recognition: "state of the art"," *IJCSI International Journal of Computer Science Issues*, vol. 9, 02 2013.

[8] M. Wang and W. Deng, "Deep face recognition: A survey," *Neurocomputing*, vol. 429, pp. 215–244, mar 2021.

[9] S. Emami and V. Suciu, "Facial recognition using opencv," *Journal of Mobile, Embedded and Distributed Systems*, vol. 4, 03 2012.

[10] A. Benton, "Facial recognition 1990," *Cortex*, vol. 26, no. 4, pp. 491–499, 1990.

[11] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," vol. 1, pp. I–511, 02 2001.