The Journal of Multidisciplinary Research

(TJMDR)

Content Available at www.saap.org.in Online ISSN: 2583-0317

Review Article Open Access

Live video face recognition using raspberry pi

Mathe Rama Krishna[1], Ambati Partha Sivanand[2]

[1]Asst. Professor, Department of CSE, Adikavi Nannaya University Rajahmundry

[2]Department of Computer Science, University College of Engineering, AdikaviNannaya University

Article History: Received: 22 Sept, 2021 Revised: 27 Nov, 2021 Accepted: 02 Dec, 2021

Abstract

Face Detection and Recognition is a challenging area of computer vision that involves in identification of human faces. Face recognition has a lot of applications in real-time like in security, accessibility, and even in payments. Since the number of applications increases, there is a need of more efficient, reliable and low-cost face recognition system in today's world. The aim of the paper is to develop such type of face recognition system using Raspberry Pi device. The setup consists of a Raspberry Pi 4 Model B device with a camera module attached. The camera is used to take the input images of any person and store them to train our model. The proposed model makes use of OpenCV, TheHaar Cascadeclassifiers for face detection and the HOG(Histogram of Gradient Descent) + SVM classifiers for real time face recognition. After vigorous testing the overall complexity and accuracy of the model the system is proved to be efficient and robust.

Keywords: Haar Cascade, HOG, Face Recognition, Open CV.

Corresponding Author

Ambati Partha Sivanand

INTRODUCTION

In the present days, as technology is getting more advanced every day, images or live camera has becomethe most convenient and preferable mode of expression for almost everybody. Various photos are uploaded everydayon different clouds and social media platforms. Retrieving these images and organizing them is a challenging task.

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.

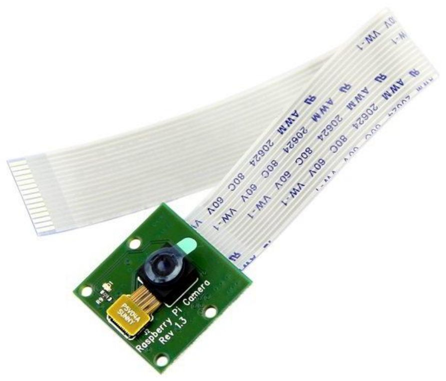
Face detection and recognition has been a topic of active research for a long period of time. There are well defined and developed technologies are available for human face detection and recognition as well in a photo or real time using web camera. A very easy example of it is the face recognition system in every smartphone that recognize 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, I have built a Face recognition system using a Raspberry Pi device along with a camera module attached to it. In this I used the Haar Cascade classifier for face detection and the HOG(Histogram of Gradient Descents) algorithm with SVM classifiers for face recognition .The aim of the project is to achieve a low-cost reliable system which can be used for variety of applications.

Let's discuss about the two main hardware components that are used in the project.

i. Raspberry Pi 4 Model B
ii. Pi Camera module

Raspberry Pi can called a mini-computer which is of the same size as that of a credit card. The Raspberry Pi is a small and affordable single-board computer that we will use to design and develop practical IoT devices while learning about programming language and computer hardware systems. In addition, you will learn how to set up the Raspberry Pi, get a Linux operating system running, and write and execute some basic Python code on the Raspberry Pi. You will also learn how to use Python-based IDE (integrated development environments) for the Raspberry Pi and how to trace and debug Python code on the device. The above figure depicts the Raspberry Pi 4 model B which is used in this project.



Pi Camera module is a camera that is used to take pictures and high definition video.It is attached to the raspberry pi. It's features are 5 MP resolution, HD recording upto 1080P @30 FPS and is CSI interface enabled. Apart from these we also use an adapter to givepower to the raspberry pi and a USB cable which is connected to the raspberry pi to the monitor of the system and also we can connect a keyboard and mouse to our raspberry pi device.

II. LITERARY SURVEY

A. *Facial Recognition and its origin*

Facial recognition is a method of identification of a person by their face in a computer application usually using machine learning. Face recognition systems uses some of the salient features such as eyes, eyebrows, nose, chin, color of skin, etc. of a person's face to identify a face. These features are thenturned into mathematical data and compared with the facial database.The earliest pioneers of facial recognition were Woody Bledsoe, Helen Chan Wolf and Charles Bisson. In 1964 and 1965, Bledsoe, along with Wolf and Bisson began work using computers to recognize the human face.

Due to the funding of the project originating from an unnamed intelligence agency, much of their work was never published. However, it was later revealed that their initial work involved the manual marking of various "landmarks" on the face such as eye centers, mouth etc. These were then mathematically rotated by a computer to compensate for pose variation. The distances between landmarks were also automatically computed and compared between images to determine identity.

These earliest steps into Facial Recognition by Bledsoe, Wolf and Bisson were severely hampered by the technology of the era, but it remains an important first step in proving that Facial Recognition was a viable biometric.

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. *Various algorithms used*

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

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. 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) **Local Binary Pattern (LBP)**: This technique is used for feature extraction. The face is divided into several small regions from which histograms are extracted.These histograms are then concatenated into a single enhanced histogram which can then be used to match faces as demonstrated in the paper "Face Recognition Based On Local Binary Pattern." (2018) written by Gondole, Devendra, and P. A. Salunkhe. This algorithm is more robust to face conditions but it is not feasible in real time environments.

3) **Principal Component Analysis (PCA):** Human recognition with the help of PCA was done by Turk and Pent land [8]. The recognition technique, called Eigen face technique defines an area 1 International Journal of Computer Applications (0975 - 8887) Volume 176 - No.33, June 2020 which reduces the depth of the authentic data space. This reduced knowledge area is used for recognition.

4) **Genetic Algorithms**: One of the main challenges in face recognition is feature selection. In fact, it is a global optimization problem in machine learning. It is used to remove the number of features and irrelevant, noisy, redundant data in order to improve efficiency and accuracy. Methods based on genetic algorithms have been proposed [8, 9] which help to optimize the search strategies for feature selection. This can be particularly useful in real time applications. They have been used in tandem with some other techniques like Principal Component Analysis and Discrete Cosine Transform to achieve up to 99% [8] accuracy in face detection.

III. METHODOLOGY

A. System used

As discussed above the face recognition system works in mainly three steps i.e. face detection, feature extraction and face recognition. Face detection and feature extraction are done simultaneously . As we are using live video capture forthe usersface detection, we follow the system discussed in the paper 'Image-based Face Detection and Recognition "State of the art".

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.

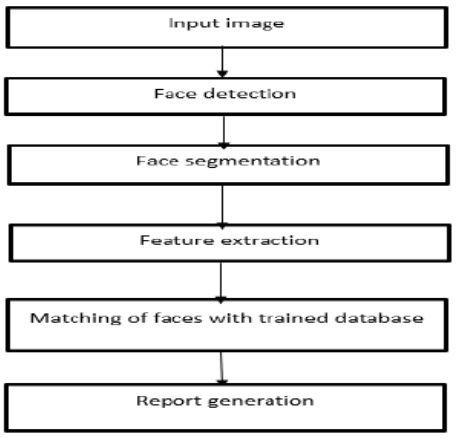


Fig. 2. Configuration of a generic face recognition System

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.

B, Algorithms

1) **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

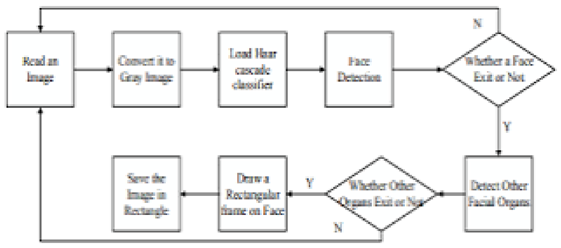

Read an Image
Convert it to Gray Image
Load Haar cascade classifier
Face Detection
Whether a Face Exit or Not
N
Y
Detect Other Facial Organs
Whether Other Organs Exit or Not
Y
N
Draw a Rectangular frame on Face
Save the Image in Rectangle

Fig. 1. Haar Cascade Flow Chart

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 theimage. Only those image regions, called sub-windows, that pass through all the stages of the detector are considered to contain the target object . In this project we are using Haar Cascade due to its ease and efficiency.

2) **HOG (Histogram Oriented Gradient Descent) :**The histogram of oriented gradients (HOG) is a feature descriptor usedin image processing for the purpose of recognition. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spacedcells and uses overlapping local contrast normalization forimproved accuracy. The essential thought behind the histogram of oriented gradients descriptor is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled. The descriptor is the concatenation of these histograms. For improved accuracy, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block. This normalization results in better invariance to changes in illumination and shadowing. The final step in object recognition using histogram of oriented gradient descriptors is to feed the descriptors into some recognition system based on supervised.

3)Support Vector Machine (SVM) : The support vector machine (SVM) classifier is a binary classifier which looks for an optimal hyper plane as a decision function. Once trained on images containing some particular object, the SVM classifier can make decisions regarding the presence of an object, such as a human, in additional test images. More formally, a support vector machine constructs a hyper plane or set of hyper planes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyper plane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. In this way both we use both HOG and SVM for face recognition.

C. Implementation and Working

<u>**Part 1**</u>: Install Dependencies for Raspberry Pi

In this step, we will install OpenCV, Imutils, and Face_recognition module and temporarily modify our swapfile to prepare our Raspberry Pi for machine learning and facial recognition.OpenCV is an open-source software library for processing real-time image and video with machine learning capabilities.

We will use the Python Face_recognition package to compute the bounding box around each face, compute facial embedding, and compare faces in the encoding dataset.Imutils is a series of convenience functions to expedite OpenCV computing on the Raspberry Pi.

1. Plug in your webcam into one of the USB ports of your Raspberry Pi.
2. Boot your Raspberry Pi with the micro SD card.
3. Open a Terminal. Install Open CV into your raspberry pi by using the command :

git clone https://github.com/opencv/opencv.git
git clone https://github.com/opencv/opencv_contrib.git

4. Before installing OpenCV files make sure CMake is present in your system.
5. Install face_recognition. This can be done by using the command :

pip install face-recognition

6. Install imutils by using the following command.

pip install impiputils

<u>**Part 2**</u>: Train the model for Face Recognition

In this section, we will focus on collecting image data and training our model to detect faces.

1. Open a new terminal on your Pi by pressing Ctrl-T.
2. Now we need to create three python files i.e. one for taking images and to store in the raspberry pi i.e. the enrol.py file and other to train the model based on the images i.e. the face_train.py file and the last one is for real-time face recognition i.e. the Face_rec.py. Put all three files in a single folder.

3. Now create a dataset folder in which the images taken are stored and are trained for recognition.
4. Enter your first name for the name of your newly created folder.This name is mentioned in the enrol.py file so that all the images taken will be stored here.
5. Click OK to finish creating your folder.
6. Now in the file manager, navigate to facial_recognition folder and open enrol.py file in Geany IDE.
7.Now run the enrol.py code in Geany. A new window will open with a view of your webcam. (It generally takes 9-10 seconds to pop up).
8. Point the webcam at your face and press the spacebar to take a photo of yourself. Each time you press the spacebar you are taking another photo. Take atleast 10 photos of your face at different angles (turn your head slightly in each photo).

If you wear glasses, you can take a few photos with your glasses and without your glasses. Hats are not recommended for training photos. These photos will be used to train our model. Press Esc when you have finished taking photos of yourself.

9. Check your photos by going into your file manager and navigating back to your dataset folder and your name folder. Double-click on a single photo to view. Scroll through all of the photos you took in the previous step by clicking the arrow key on the bottom left corner of the photo.
10. Repeat steps 5 through 10 to add someone else in your family.

Now that we have put together our dataset, we are ready to train our model.

11. In a new terminal, navigate to facial_recognition by typing:

cd facial_recognition

12. Run the command to train the model by entering:

python face_train.py

The steps involved in the face_train.py file are :

- Dataset: face_train.py will analyze photos within the dataset folder. Organize your photos into folders by person's name. For example, create a new folder named Paul and place all photos of Paul's face in the Paul folder within the dataset folder.
- Encodings: face_train.py will create a file named encodings.pickle contains the criteria for identifying faces in the next step. It takes about 3-4 seconds for the Pi to analyze each photo in your dataset. For a dataset with 20 photos, it will take about 1.5 minutes for the Pi to analyze the photos and build the encodings.pickle file.
- Detection Method: We are using the HOG + SVM detection method.

Now let's test the model we just trained.

16. Run the command to test the model by typing:

python facial_rec.py

In a few seconds, your webcam view should open up. Point the webcam at your face. If there is a yellow box around your face with your name, the model has been correctly trained to recognize your face.

Fig . Experiment setup

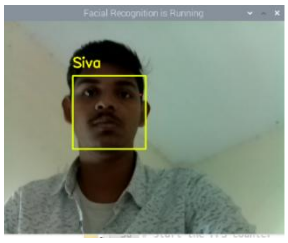
IV. EXPERIMENT AND RESULTS

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.

These are the images used to train the model.

B. Results



The system recognized all the trained faces. The average accuracy obtained over a duration of 60 seconds was 72.01% with a maximum value of 98.53% at a frame rate of 2 FPS. These numbers indicate that the Raspberry Pi can be used for simple real-time applications despite its modest hardware capability.

C. Conclusion

Image and video classification is a challenging area of research in the computer vision field. The are many applications for image and video analysis in real-time. The recent focus of research for image and video analysis is the use of machine learning and deep learning models for high resolution images with Iot devices. In this paper, we have a presented a low cost and reliable face recognition system using raspberry pi. This type of systems can be used as biometric security systems in schools and colleges. It can be also used for crowd monitoring, assistive devices and many more. In this way the field of face recognition is growing at a tremendous pace right now and there is a need for a low cost and reliable systems.

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] Zhi, Hui, and Sanyang Liu. ”Face recognition based on genetic algorithm.” Journal of Visual Communication and Image Representation 58 (2019): 495-502.

[9] Moussa, Mourad, MahaHmila, and Ali Douik. ”A novel face recognition approach based on genetic algorithm optimization.” Stud. Inform. Control 27.1 (2018): 127-134. A. Benton, “Facial recognition 1990,” Cortex, vol. 26, no. 4, pp. 491–499, 1990.

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

[11] Gondole, Devendra, and P. A. Salunkhe. ”Face Recognition Based On Local Binary Pattern.” (2018).