International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 05 Issue: 05 | May - 2021 ISSN: 2582-3930

OPENCV FOR EMPLOYEE ATTENDANCE AND MANAGEMENT

Sanjana Kulkarni, Satyam Singh, Rajan Yadav, Prof Madhura Phadke

BE Computer Engineering ,Datta Meghe College of Engineering ,Navi Mumbai,India

BE Computer Engineering ,Datta Meghe College of Engineering ,Navi Mumbai,India

BE Computer Engineering ,Datta Meghe College of Engineering ,Navi Mumbai,India

Assistant Professor, Department of Computer Engineering,Datta Meghe College of Engineering ,Navi Mumbai,India

---***---

Abstarct : Face is the crucial part of the human body that uniquely identifies a person. Using the face characteristics as biometric, the face recognition system can be implemented. The most demanding task in any organization is attendance marking. In traditional attendance system, the students are called out by the teachers and their presence or absence is marked accordingly. However, these traditional techniques are time consuming and tedious. In this project, the Open CV based face recognition approach has been proposed. This model integrates a camera that captures an input image, an algorithm for detecting face from an input image, encoding and identifying the face, marking the attendance in a spreadsheet and converting it into PDF file. The training database is created by training the system with the faces of the authorized students. The cropped images are then stored as a database with respective labels. The features are extracted using HOG algorithm.

I Introduction

Attendance maintenance is one of the most significant functions .Every institution has their own method of maintaining attendance .In today's world maintaining a written attendance is a tedious process. Hence building an online attendance system is important to save up some time and to digitize the traditional methods. A facial recognition system is a computerized biometric software which is suited for determining or validating a person by performing comparison on patterns based on their facial appearances.Face recognition systems have upgraded appreciably in their management over the recent years and this technology is now vastly used for various objectives like security and in commercial operations as well. Face recognition is a powerful field of research which is a computer based digital technology and usually implemented using OpenCV. The face recognition system analyses video feed frame by frame for detection of faces. Then a predictor is used to identify the facial landmarks in the faces detected using a predefined template. Using this template, the faces are aligned to increase the accuracy of the model. Now the unique embeddings of these aligned faces are generated. These embeddings are biometrics of the face and hence unique. These embeddings are classified using the classifier. This classifier is trained on the images collected from the users beforehand to ensure the smooth workflow of the program.

II Overview

Humans have always had an innate ability to recognize and distinguish between faces. An attendance system is a biometric technology used for mapping facial features and patterns for the purpose of identity storage and verification. It is being widely used to improve access and security, allows payments to be processed without physical cards. It also enables criminal identification, strengthening the pillars of law and enforcement. It also has started breaking natural barriers by helping blind or low vision users recognize faces and their surroundings. Face recognition being a biometric technique implies determination if the image of the face of any particular person matches any of the face images that are stored in a database. This difficulty is tough to resolve automatically because of the changes that several factors, like facial expression, aging and even lighting can affect the image. Facial recognition among the various biometric techniques may not be the most authentic but it has various advantages over the others.

III Image Processing with OPENCV

OpenCV is an open-source library written in C++ that provides a set of programming functions that cater to real-time computer vision by making use of classic and state of the art Machine Learning and Computer Vision algorithms. Its applications include object detection, object tracking, 3D model extraction, face detection and recognition, scenery recognition, etc. OpenCV is used extensively in this project for webcam stream processing and image processing required for face detection, feature extraction, and alignment.

a. *OpenCV BASICS*

OpenCV can be installed by building from source which can be downloaded from OpenCV's website (Installation - https://opencv.org/releases/). Once it is downloaded, it can be imported using the command: import cv2. In order to load an image in OpenCV, the imread() method is used, in which the path of the image file is provided as a parameter. An image, once loaded, can be displayed using the imshow() method. Images are loaded as numpy arrays, and hence their dimensions can be retrieved using the shape attribute of numpy arrays. This provides another advantage - cropping images and extracting regions of interest is reduced to array slicing. Images can be resized using the resize() method.

Rectangles, circles and text can be drawn onto an image using the rectangle(), putText() and circle() methods.

OpenCV loads images in BGR format which is not always desirable. Colorspace conversions to Grayscale and RGB formats is achieved easily using the cvtColor() method. All of these methods are proposed to be used throughout the project for image manipulation.

b. *Working with Video Streams*

OpenCV provides VideoCapture and VideoWriter classes to capture videos and save them on disk, respectively. In this project, VideoCapture objects have been used to work with webcam streams. The VideoCapture() constructor takes one argument, which can either be a number, specifying the index of the camera device, or the path to a video file. Once a VideoCapture object is created, it can be processed frame by frame, allowing us to apply the same image processing techniques that we use for static images. The video capture can be released with the release() method.

c. *Imutils Library*

Imutils is a python library that provides a series of convenience functions that make basic image processing tasks like translation, resizing, rotation and video stream processing a lot easier. OpenCV and imutils is hence used in this project to capture videos via web camera, process them frame by frame, and transform those frames into a format that acts as a suitable input for face detectors, shape predictors, aligners, and models that extract embeddings.

IV FACE DETECTION AND ALIGNMENT

Face detection is the technology of recognizing human faces in a photo, video or live frame. Face detection can be said as a specific case of object-class detection. Face detection is the first step in the process of facial recognition.

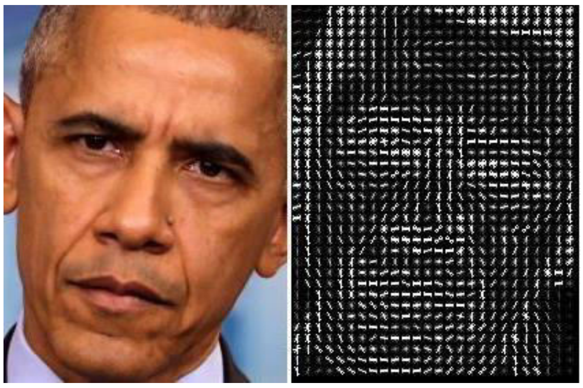
Face alignment, that can be thought of as data normalisation, is a part of image preprocessing, wherein, after detecting the face, the image is cropped, rotated, scaled and adjusted so as to provide a normalized input to the face classifier

Histogram of Oriented Gradients (HOG) detector.

HOG is a feature descriptor used in computer vision and image processing for the purpose of object detection. It works on the concept of finding gradients along the x and y-axis. These gradients are added vectorially and information such as magnitude and direction of the gradients are extracted forming edges in the image. Now the whole gradient image is divided into 8*8- dimension matrix

i.e. carrying 64 gradient vectors. Each matrix contains 9 bins and each bin has a range of 40 degrees to allocate the gradient vector lying in that range. After each gradient vector has been allocated a bin, a histogram of this matrix is formed which determines the overall direction and magnitude of the gradient.

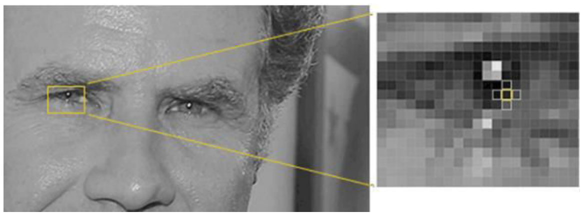
Luminosity plays a great role in the proper detection of gradients. To overcome this failure the value of the histogram is obtained in the red, green and blue frame and then normalized making it lighting invariant. Finally, the normalized histogram is compared with its default histogram for faces to identify a face in an image.



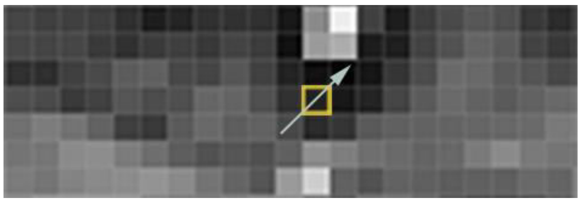
The following code is used to initialize the HOG based face detector: hog_face_detector = dlib.get_frontal_face_detector().

Steps:
1:To find faces in an image, we'll start by making our image black and white because we don't need colour data to find faces

2: Then we'll look at every single pixel in our image one at a time. For every single pixel, we want to look at the pixels that directly surround it.

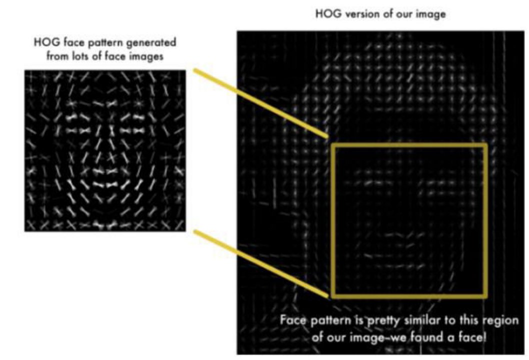


3: Our goal is to figure out how dark the current pixel is compared to the pixels directly surrounding it. Then we want to draw an arrow showing in which direction the image is getting darker .



4: Generalization.

5:Detection :


HOG version of our image
HOG face pattern generated from lots of face images
Face pattern is pretty similar to this region of our image-we found a face!

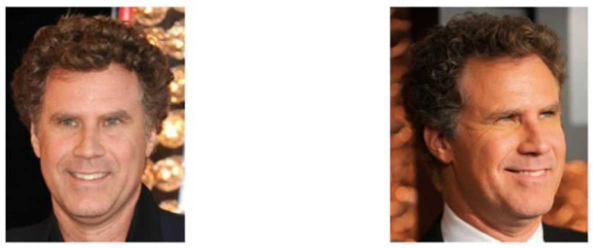
V FACIAL RECOGNITION

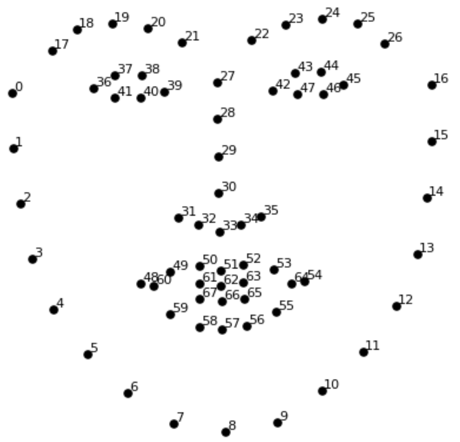
A facial recognition system typically works by extracting face encodings from faces and comparing them with an existing database of known faces to find a match. Given an unknown image, the process of facial recognition comprises detecting faces in that image, aligning them, extracting feature vectors from aligned faces and classifying each feature vector as belonging to one class from a set of classes.

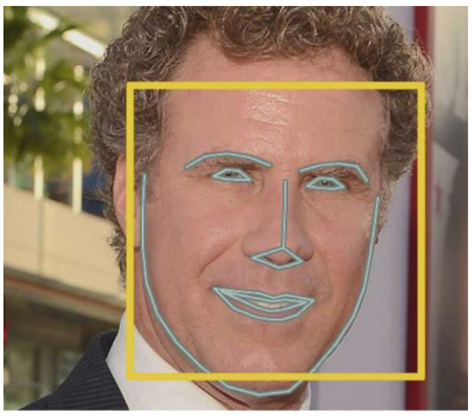
Landmark Detection Using DLIB'S 68pt Shape Detector

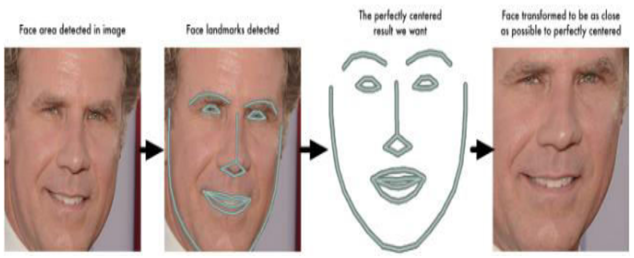
DLIB is an open-source machine learning and computer vision library written in c++. Its python API combines fast and robust c++with easy to use python making it able to archive many practical life tasks.

The good thing is that the same DLIB framework can be used to train a shape predictor on the input training data — this is useful if you would like to train facial landmark detectors or custom shape predictors of your own for example hands predictor or coin predictor etc.


Humans can easily recognize that both images are of Will Ferrell, but computers would see these pictures as two completely different people.




The above figure shows the 68pt marking of Will Pharell's phase .


Face area detected in image
Face landmarks detected
The perfectly centered result we want
Face transformed to be as close as possible to perfectly centered

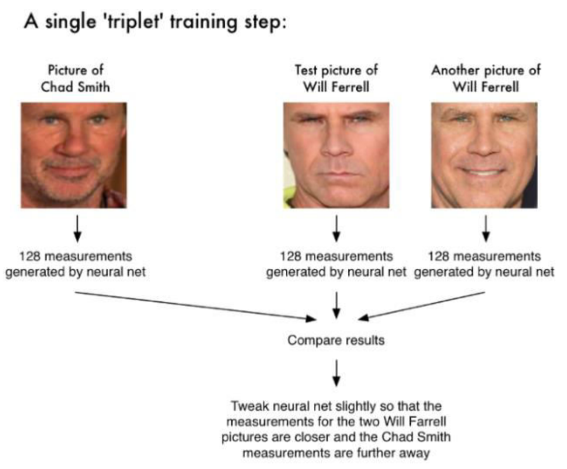
Extracting 128-d Embeddings

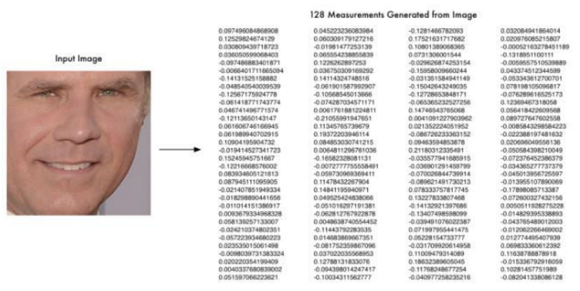
The training process works by looking at 3 face images at a time:

. Load a training face image of a known person

. Load another picture of the same known person

. Load a picture of a totally different person

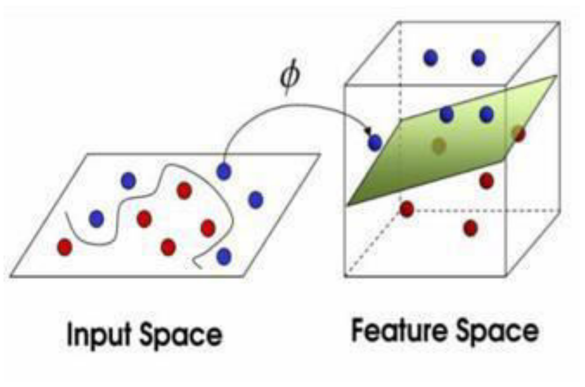

A single 'triplet' training step:
Picture of Chad Smith
Test picture of Will Ferrell
Another picture of Will Ferrell
128 measurements generated by neural net
128 measurements generated by neural net
128 measurements generated by neural net
Compare results
Tweak neural net slightly so that the measurements for the two Will Farrell pictures are closer and the Chad Smith measurements are further away

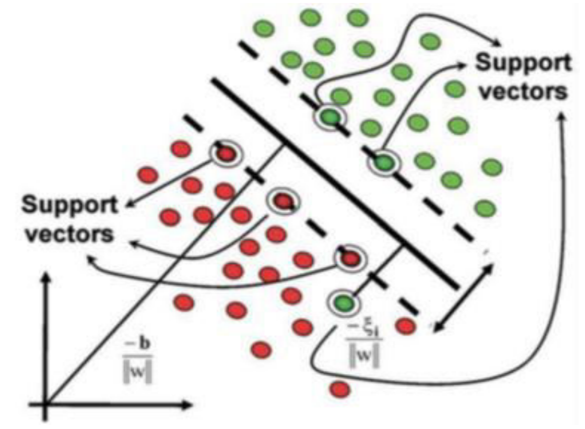


0.097496084868908	0.045223236083984	-0.1281466782093	0.032084941864014
0.12529824674129	0.060309179127216	0.17521631717682	0.020976085215807
0.030809439718723	-0.01981477253139	0.10801389068365	-0.00052163278451189
0.036050599068403	0.065554238855839	0.0731306001544	-0.1318951100111
-0.097486883401871	0.1226262897253	-0.029626874253154	-0.0059557510539889
-0.0066401711665094	0.036750309169292	-0.15958009660244	0.043374512344599
-0.14131525158882	0.14114324748516	-0.031351584941149	-0.053343612700701
-0.048540540039539	-0.061901587992907	-0.15042643249035	0.078198105096817
-0.12567175924778	-0.10568545013666	-0.12728653848171	-0.076289616525173
-0.061418771743774	-0.074287034571171	-0.065365232527256	0.12369467318058
0.046741496771574	0.0061761881224811	0.14746543765068	0.056418422609568
-0.12113650143147	-0.21055991947651	0.0041091227903962	0.089727647602558
0.061606746166945	0.11345765739679	0.021352224051952	-0.0085843298584223
0.061989940702915	0.19372203946114	-0.086726233363152	-0.022388197481632
0.10904195904732	0.084853030741215	0.09463594853878	0.020696049556136
-0.019414527341723	0.0064811296761036	0.21180312335491	-0.050584398210049
0.15245945751667	-0.16582328081131	-0.035577941685915	-0.072376452386379
-0.12216668576002	-0.0072777755558491	-0.036901291459799	-0.034365277737379
0.083934605121613	-0.059730969369411	-0.070026844739914	-0.045013956725597
0.087945111095905	0.11478432267904	-0.089621491730213	-0.013955107890069
-0.021407851949334	0.14841195940971	0.078333757817745	-0.17898085713387
-0.018298890441656	0.049525424838066	0.13227833807468	-0.072600327432156
-0.011014151386917	-0.051016297191381	-0.14132921397686	0.0050511928275228
0.0093679334968328	-0.062812767922878	-0.13407498598099	-0.014829395338893
0.058139257133007	0.0048638740554452	-0.039491076022387	-0.043765489012003
-0.024210374802351	-0.11443792283535	0.071997965441475	-0.012062266469002
-0.057223934680223	0.014683869667351	0.05228154733777	0.012774495407939
0.023535015061498	-0.081752359867096	-0.031709920614958	0.069833360612392
-0.0098039731383324	0.037022035568953	0.11009479314089	0.11638788878918
0.020220354199409	0.12788131833076	0.18632389605045	-0.015336792916059
0.0040337680839002	-0.094398014247417	-0.11768248677254	0.10281457751989
0.051597066223621	-0.10034311562777	-0.040977258235216	-0.082041338086128

Classification Using SVM

Support Vector Machine is a linear model for classification. It can solve many linear and non-linear practical problems. The algorithm creates a line or a hyperplane which separates the data into classes. If the classes are inseparable in input space, the SVM plots these data points in a higher dimension using additional parameters. This higher dimensional space is called the feature space. The feature space for face encodings is 128-dimensions. The test image is classified to the new cluster in the feature space.

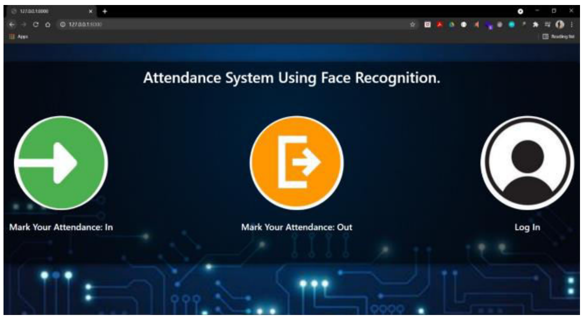




VI WEB DEVELOPMENT

The project envisions the creation of a web application that would work as an attendance system that uses the facial recognition model described in previous sections. For the development of this application, Django would be used. Django is a high-level Python-based, an open-source web framework that allows the creation of websites with ease. It follows the Model-Template-View (MTV) Architecture (elaborated upon below). Along with being secure, portable, scalable, and versatile, it takes care of the intricacies and hassle of web development, so that the developer can focus solely on writing the application.

Main Dashboard


Attendance System Using Face Recognition.
Mark Your Attendance: In
Mark Your Attendance: Out
Log In

Admin Dashboard

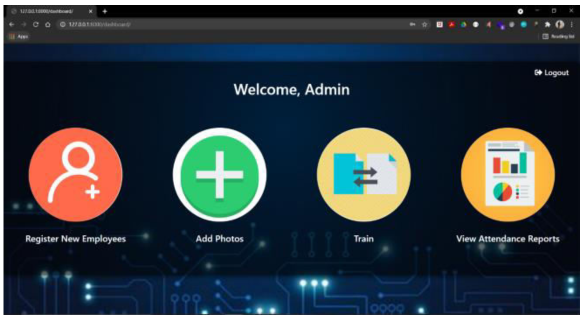


Image being Captured

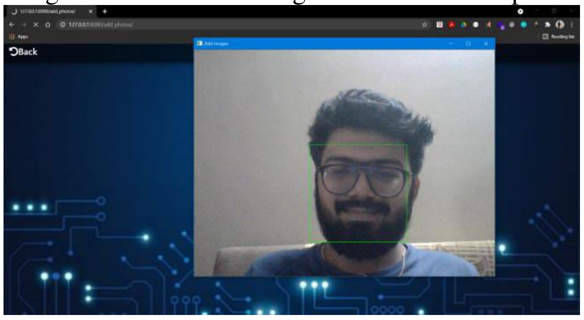
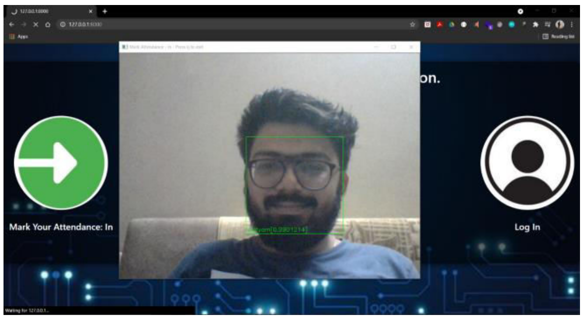


Image being recognised



References

[1] A. J. Goldstein, L. D. Harmon, and A. B. Lesk, "Identification of Human Faces," in Proc. IEEE Conference on Computer Vision and Pattern Recognition, vol.59, pp 748 - 760, May 1971

[2] M. A. Fischler and R. A. Erschlager, " The Representation and Matching of Pictorial Structures," IEEE Transaction on Computer, vol. C-22, pp. 67-92, 1973

[3] S. S. R. Abibi, "Simulating evolution: connectionist metaphors for studying human cognitive behaviour," in Proceedings TENCON 2000, vol. 1 pp 167-173, 2000

[4] Y. Cui, J. S. Jin, S. Luo, M. Park, and S. S. L. Au, "Automated Pattern Recognition and Defect Inspection System," in proc. 5th International Conference on Computer Vision and Graphical Image, vol. 59, pp. 768-773, May 1992

[5] Abdol hossein Fathi, Pendar Alirezazadeh, FardinAbdali-Mohammadi. (2016) "A new Global-Gabor-Zernike feature descriptor and its application to face recognition" Journal of Visual Communication and Image Representation 38: 65-72.

[6] Z.H.D.Eng, Y.Y.Yick, Y Guo, H.Xu, M.Reiner, T.J.Cham, S.H.A.Chen. (2017) "3D faces are recognized more accurately and faster than 2D faces, but with similar inversion effects" Vision Research 138: 78-85

[7] Ayan Seal, Debotosh Bhattacharjee, Mita Nasipuri.(2016) ""Human face recognition using random forest based fusion of à-trous wavelet transform coefficients from thermal and visible images" AEU- International Journal of Electronics and Communications 70(8):

1041-1049.

[8] Md. Zia Uddin, Mohammed, Mehedi Hassan, Ahmad Almogren, Mansour Zuair, Giancarlo Fortino, Jim Torresen. (2017) "A facial expression recognition system using robust face features from depth videos and deep learning" Computers & Electrical Engineering Available online 29 in press.

[9] Thibault Napoleon, Ayman Alfalou. (2017),"Pose invariant face recognition: 3D model from single photo", Optics and Lasers in Engineering 89: 150-161.