Attendance Monitoring System using Face Recognition

Siddhant Medar[1], Jayendra Deshmukh[2], Harshal Mahadik[3], Shantanu Ingale[4]

[1,2,3,4]*Student, Dept. of Electronics & Telecommunication Engineering, Vidyalankar Institute of Technology, Mumbai, Maharashtra, India*

---***---

Abstract - *Every individual has distinctive facial features and can be used to identify a person. Our project aims at semi-automating the existing attendance systems and introduces more transparency in the whole process of managing attendance records.*

This system achieves near real-time rates using image processing algorithms. The facial-features are studied and used for training and testing the dataset. The system model is then used to make a prediction based on the features extracted. The proposed system accurately detects human face with the help of the features extracted in the previous steps, which reveals that the proposed system is able to efficiently work on our dataset along with accurate results.

On successful detection of the human face, the system would verify whether the face is present in the dataset and in case, a match is found then the system will mark the user as present. On occasion of detection of an unknown face, the system would display 'unknown' to indicate that the user is not registered. The attendance record is stored in the form of MS Excel sheet reports. The attendance reports can be sent over email to the concerned entity as and when required.

***Key Words*: Face Detection, LBPH recognizer, AdaBoost, Haar Cascade Classifier**

1. INTRODUCTION

Attendance is being monitored in Educational institutes, Government offices, and IT companies to maintain a record of its students or employees. There are different ways to carry out this process like biometric attendance monitoring, barcode scanning or keeping a register book to record attendance. Out of these mentioned above, the most commonly used method in most of the educational institutes is roll call list, this method is time consuming and is also not reliable, as attendance can be marked/ signed by some individual as well.

The main goal of our project is to semi-automate the existing attendance monitoring system and increase transparency, reliability and ease the whole process of monitoring attendance in various sectors by using the fundamentals of Computer Vision field. This project aims at applying computer vision algorithms to semi-automate the existing attendance systems and introduces more transparency and reliability in the process of managing attendance records.

The ideology is to provide the educational institutes, an efficient means to monitor the attendance of its students. Report generation in MS Excel and sharing reports with others over e-mail. The system automatically generates the attendance reports and stores them on the host device in .xlsx format. It also provides an option to send the attendance reports to anyone by simply entering their e-mail address into the respective dialog box provided in the GUI program.

2. METHOD

The attendance monitoring system consists of three phases:

1. Face Detection and Data Gathering
2. Train the Recognizer
3. Face Recognition

2.1. Face Detection and Data Gathering

In this phase, first we take custom inputs such as name, e-mail ID and Student ID No. from the user and then run a takeImages() function to capture the image of the user and also write those details into the StudentDetails.csv file. In takeImages() function, we first load the Haar Cascade Classifier and also create a VideoCapture object using OpenCV for image capturing purpose. Then, the image is captured, the face is extracted from the image and is stored inside the Training Image folder after converting it into grayscale for image processing purpose.

2.2. Train the Recognizer

In this second phase, the system is trained on the image captured in the previous phase. The LBPHFaceRecognizer's train() function is used for this purpose and the training data is stored in a "Trainner.yml" file. Hence, the final output of this phase is the .yml file.

2.3. Face Recognition

In this last phase, we first load the LBPHRecognizer and also the "Trainner.yml" generated in previous step. Then, the facial features of the subject in question are captured and are compared with all the features present inside the "Trainner.yml" file. In case the distance between the subject in question and feature in our record is less than a set threshold, then a match is found then the system identifies the user by displaying the Student ID and the name. Else if no match is found, then "Unknown" string is displayed by the system and will be prompted to register as a new user. In this phase, few additional operations are also carried out, the attendance report is generated at the end and also if an email address is provided then the same is also sent over e-mail.

3. CAPTURING OF FACIAL FEATURES USING HAAR CASCADE

Reference [2] shows that face detection has been improved in terms of speed with the application of haar-features with the contribution of the Viola-Jones object detection framework. Implementations of this framework, such as OpenCV, provide different face classifiers created by authors that used different datasets into their training. The performance and reliability of these classifiers vary a lot evaluated the performance of some classifiers and also tested their accuracy.

Historically, working with only image intensities (i.e., the RGB pixel values at each and every pixel of image) made the task of feature calculation computationally expensive. A publication by Papa Georgiou discussed working with an alternate feature set based on Haar wavelets instead of the usual image intensities. Paul Viola and Michael Jones adapted the idea of using Haar wavelets and developed the so-called Haar-like features. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.

This difference is then used to categorize subsections of an image. For example, with a human face, it is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore, a common Haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object (the face in this case). [5]

3.1. Fast computation of Haar-like features

One of the contributions of Viola and Jones was to use summed-area tables, which they called integral images. Integral images can be defined as two-dimensional lookup tables in the form of a matrix with the same size of the original image. Each element of the integral image contains the sum of all pixels located on the up-left region of the original image (in relation to the element's position). [2] This allows to compute sum of rectangular areas in the image, at any position or scale, using only four lookups:

$$\mathbf{Sum = I\,(C) + I\,(A) - I\,(B) - I\,(D)}$$

Where points A, B, C, D belong to the integral image I, as shown in the figure.

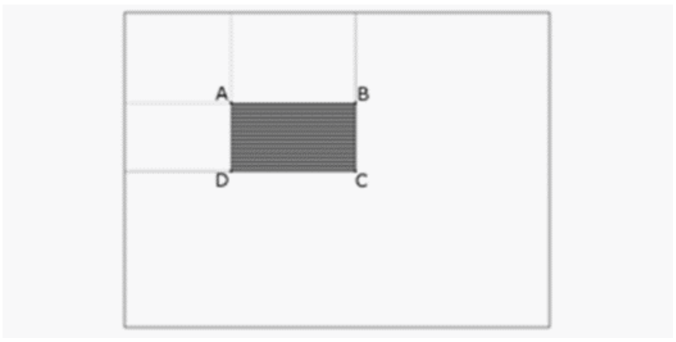


Figure 1. Finding the sum of the shaded rectangular area [5]

4. LBPH FACE RECOGNIZER

Local binary patterns (LBP) is a type of visual descriptor used for classification in computer vision. LBP is the particular case of the Texture Spectrum model proposed in 1990. LBP was first described in 1994. It has since been found to be a powerful feature for texture classification; it has further been determined that when LBP is combined with the Histogram of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets. A comparison of several improvements of the original LBP in the field of background subtraction was made in 2015 by Silva. A full survey of the different versions of LBP can be found in Bowman's.

CONCEPT

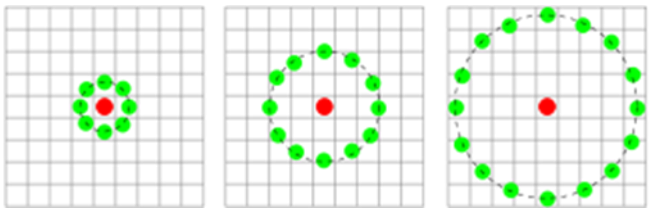


Figure 2. Three neighborhood examples used to define a texture and calculate a local binary pattern (LBP) [4]

The LBP feature vector, in its simplest form, is created in the following manner:

- Divide the examined window into cells (e.g. 16x16 pixels for each cell).
- For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left-middle, left-bottom, right- top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
- Where the center pixel's value is greater than the neighbor's value, write "0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).

- Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center). This histogram can be seen as a 256-dimensional feature vector.
- Optionally normalize the histogram.
- Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

The feature vector can now be processed using the Support vector machine, extreme learning machines, or some other machine learning algorithm to classify images. Such classifiers can be used for face recognition or texture analysis. [3]

A useful extension to the original operator is the so- called uniform pattern, which can be used to reduce the length of the feature vector and implement a simple rotation invariant descriptor. This idea is motivated by the fact that some binary patterns occur more commonly in texture images than others. A local binary pattern is called uniform if the binary pattern contains at most two 0-1 or 1-0 transitions. For example, 00010000 (2 transitions) is a uniform pattern, but 01010100 (6 transitions) is not. In the computation of the LBP histogram, the histogram has a separate bin for every uniform pattern, and all non-uniform patterns are assigned to a single bin. Using uniform patterns, the length of the feature vector for a single cell reduces from 256 to 59. The 58 uniform binary patterns correspond to the integers 0, 1, 2, 3,4, 6, 7, 8, 12, 14, 15, 16, 24, 28, 30, 31, 32, 48, 56, 60, 62, 63, 64, 96, 112, 120, 124, 126, 127, 128, 129, 131, 135, 143, 159,191, 192, 193, 195, 199, 207, 223, 224, 225, 227, 231, 239,240, 241, 243, 247, 248, 249, 251, 252, 253, 254 and 255. [4]

5. ANALYSIS AND RESULTS

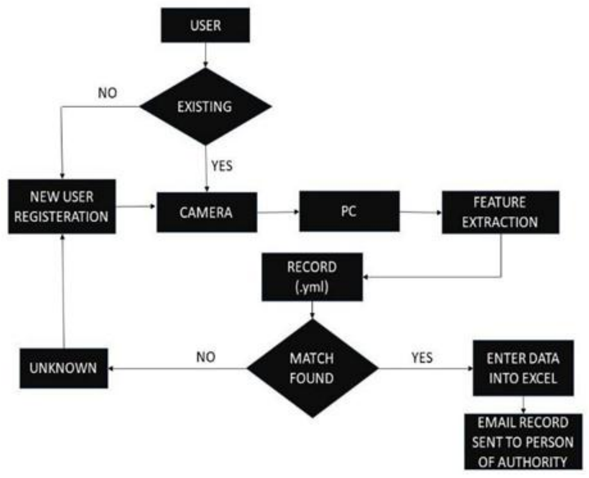


Figure 3. Flowchart

Whenever a new user is encountered by the system (on the capture window), it detects the user's facial features and compares it with all the facial feature present in the .yml file generated by the trainer phase. In case, a match is found then the system will identify the user by drawing a bounding box along with the user's ID and name. A report would be generated and a copy of the same would also be sent over e- mail if one is entered. Else, the system will identify the user as "Unknown' and would be prompted to register into the system.

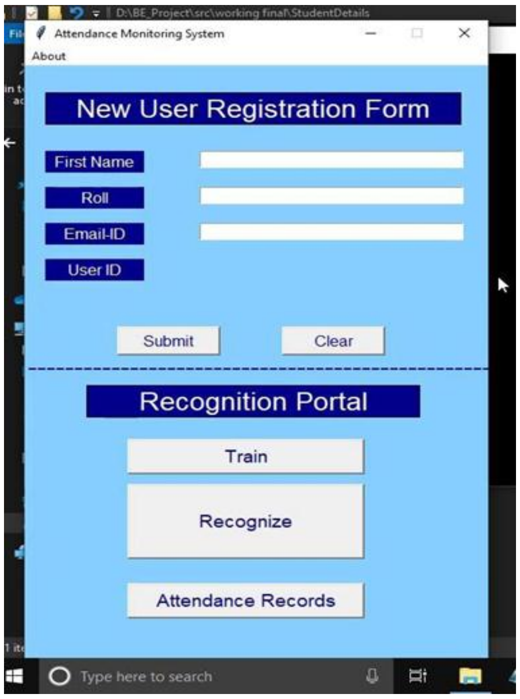


Figure 4. GUI Window

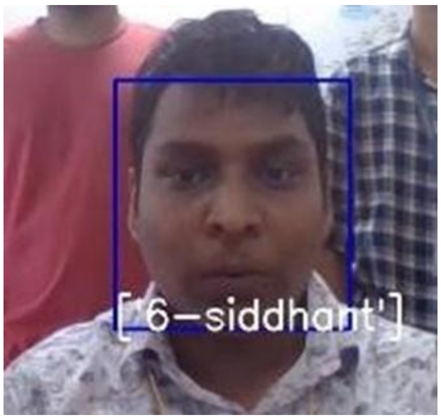
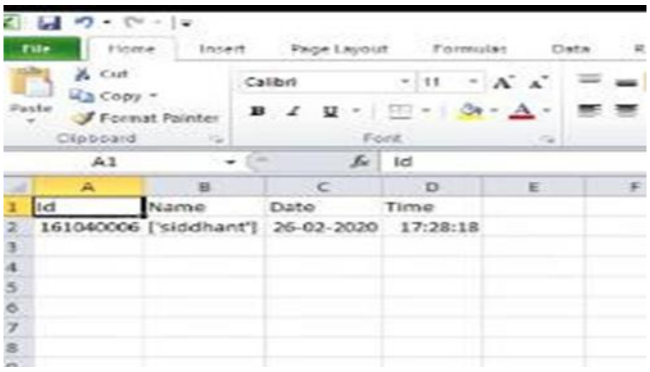


Figure 5. Face Recognition



Id	Name	Date	Time
161040006	['siddhant']	26-02-2020	17:28:18

Figure 7. Attendance Marked in Excel Sheets

Figure 8. E-mail Sent

6. CONCLUSIONS

The proposed system demonstrated makes the whole process of marking attendance can be semi-automated. It portrayed an efficiency of more than 90% when tested among a class of 70 students. Hence, it significantly improves the whole process and truly delivers what it aims for.

REFERENCES

[1] Rapid object detection using a boosted cascade of simple features - P.
[2] Viola, M. Jones, Mitsubishi Electr. Res. Labs., Cambridge, MA, USA
[3] Padilla, Rafael & Filho, Cicero & Costa, Marly. (2012). Evaluation of Haar Cascade Classifiers for Face Detection.
[4] Face Description with Local Binary Patterns: Application to Face Recognition - Timo Ahonen, Abdenour Hadid, and Matti Pietikainen
[5] https://en.wikipedia.org/wiki/Local_binary_patterns
[6] https://en.wikipedia.org/wiki/Haar-like_feature