Face Recognition Attendance System

Ayush Atul Hate

Abstract*: *Taking attendance in classrooms, especially in large college classrooms, is a tedious, time consuming task which is error prone and a nuisance to the flow of teaching. The Face Recognition Attendance System presents a merger of the novel technology of 'Facial Recognition' to help alleviate the overhead of this mundane task. This face recognition implementation aims to complete the entire task of taking attendance and updating a database in a fraction of the time normally taken using one or more pictures of the class. The Face Recognition Attendance System leverages the programming languages of Python using the OpenCV package for the main image processing task and a web technology stack of HTML, CSS, and PHP for the front end system for ease of use for the end users.

Keywords: Attendance System, Facial Recognition, OpenCV, Python

I. INTRODUCTION

The Face Recognition Attendance System is a amalgamation of Web Technology, Database Technology and Machine Learning to solve a simple problem in day to day life. Taking attendance is a mundane task which is prevalent in every educational institute. This application aims to alleviate this task by using facial recognition to detect, recognize and update attendances in databases. Marking attendance using conventional methods are prone to certain problems including errors, students giving proxies, time consuming paper-pen work, manual management, etc. This project aims to develop a system streamlining the process of taking attendance by implementing facial recognition on images of a classroom automatically updating the correct entries in a database. In educational institutes everywhere the common practice is a repetitive, error prone manual method wherein the professor calls out names and students replies "Present". Going one step further, in some business establishments, face recognition has been implemented at doorways/entrances or similar prime points of contact. At these point of contacts, an image of each individual is captured, processed and logged. This method, although efficient for a small number of people, leads to long queues and thus a bottleneck for large groups. We aim to develop on this technology to create a system that is less time consuming and more intuitive in the teachers and students perspective.

II. RELATED WORK

A. Literature Survey

Face recognition has been highlighted in many research papers throughout journals. A Review Paper on Recent Use Of Local Binary Patterns and Its Variations for Face Recognition [1] by Komal and Er. Rakesh Singh highlights the benefits and use case of Local Binary Patterns Algorithm. The other classifier used for image detection is the Haar Cascade Classifier. Haar features are described and the technique is proposed in the paper Rapid Object Detection using a Boosted Cascade of Simple Features [2] by Paul Viola and Michael Jones in 2001.

III. TECHNICAL BACKGROUND

A. Local Binary Patterns Histogram

Local Binary Patterns Histogram is an old but robust type of visual descriptor developed in 1996 for computer vision applications. There are 4 main parameters of Local Binary Patterns Histograms. They are:

- Radius: Radius (in pixels) around the central pixel.
- Neighbors: Number of neighboring points to build the circular local binary pattern
- Grid X: Number of pixels horizontally
- Grid Y: Number of pixels vertically

To train the algorithm, we use a set of 20-25 training images associated with unique individuals (recognized by the University Serial Numbers (USNs)). Given a grayscale image, consider,

- Radius = 1
- Neighbors = 8
- Grid X = 8
- Grid Y = 8

We consider a sliding window of 3x3 pixels as in Fig1.

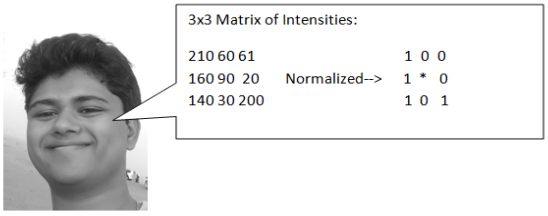

Fig1. LBPH Training

Each cell in the matrix contains the intensity of the pixel. We consider the central pixel to be the threshold and normalize the 3x3 matrix based on this threshold. In Fig1we can see that the threshold is 90. Thus, for pixels with intensity greater than 90, the normalized value is 1. Conversely, if the pixel value is less than 90, the normalized value is 0. From the normalized 3x3 matrix, we generate a binary value for that pixel window. In Fig1, this value is 10010101. This value, converted to decimal, yields 149. We replace the central pixel with the value 149 and slide the window. We repeat this process for the entire image. This Step generates an intermediate image that better represents the original image which highlights the facial features. This looks like the 'emboss' filter in many old image editing applications such as those found on old camera phones.

Revised Manuscript Received on June 05, 2019

Ayush Atul Hate, Computer Science and Engineering, Visvesvaraiya Technological University - CMR Institute Of Technology, Bangalore, India.

The intermediate image is now divided into grids of size Grid X by Grid Y (8x8 in this example). Each grid contains a histogram of intensities. All these histograms can now be concatenated to generate a histogram which is representative of the entire image. This histogram is used for recognition.

B. Haar Features and Cascading

Object detection, in this case, face detection, using Haar features is a technique proposed by Paul Viola and Michael Jones in 2001 in their paper Rapid Object Detection using a Boosted Cascade of Simple Features [2]. Machine learning methods are employed on a vast dataset of positive (with faces) and negative (without faces) images. Haar features such as those in Fig2 are detected and used on the images.

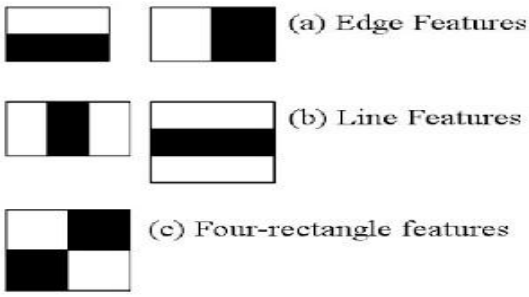

Fig2. Haar Features

Each single valued feature is calculated by subtracting the sum of pixels under the white rectangle from the sum of the pixels under the black rectangle.

Applying these features to the image yields a lot of unnecessary and useless features. Edge features are good on eye regions but irrelevant on the chin or cheek. Similarly, Vertical line features detect nose regions better than others, but, are not useful of any other region.

This method employs Adaboost to weed out irrelevant features. Adaboost works by determining threshold values for each feature type, selecting those features with the least rate of misclassifications. After each classification, weights of each weak classifier is edited and new error rates are calculated. At the end, a final classifier consisting of the weighted sum of weak classifiers is obtained.

IV. PROPOSED SYSTEM

The Face Recognition Attendance System is presented to the end user as a webapp/webpage where the teacher can upload an image of the class to be processed. The image is passed to a python script which detects, recognizes and updates the database of attendance. The webpage also provides functionalities for students and teachers to view and manage their attendance. Students can view attendance summaries and obtain useful data as to how many more classes they must attend. Teachers can update attendances individually for their particular subject and view and contact students with shortage of attendance. The proposed system core python script has 5 fundamental steps:

A. Pre-Processing:

Face detection algorithms work on greyscale images as color is not an important attribute. This step uses OpenCV methods which convert training images to greyscale. These greyscale images are used for further processing down the pipeline.

B. Face Detection:

Face detection is achieved by the use of the Local Binary Patterns Histogram algorithm and Haar cascading. This was chosen due to its simplicity and robustness to different lighting conditions

C. Model Training:

The model is trained using a written function which detects faces and assigns labels using a set of training images. We select training samples varying the angle and lighting conditions.

D. Face Recognition:

This step divides the sample image into separate images of faces to recognize. For each face, Local Binary Patterns algorithm generates a histogram which is matched with the training histograms.

E. Database Updating:

The matched label is used along with subject details to update the attendance for each detected student correctly.

The proposed system enables students and teachers to access the portal through any web enabled device. No paper work is needed and time is effectively utilized.

V. SYSTEM ARCHITECTURE

A. Architecture

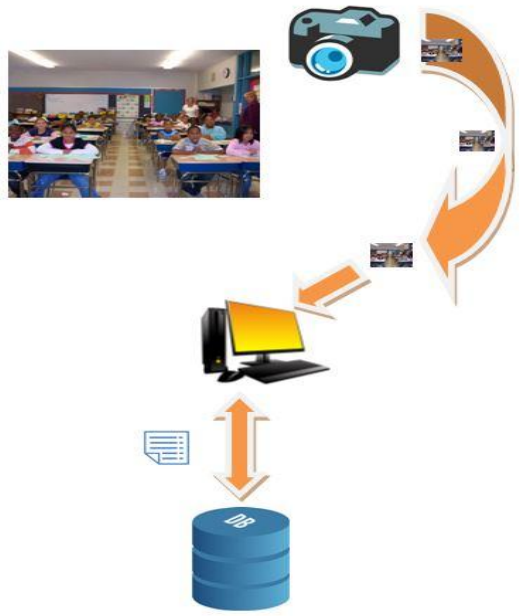

Fig3. Architecture

The architecture of the Face Recognition Attendance System is highlighted in Fig3. The Professor takes an image of the class or a part/section of the class. This photograph is uploaded to the program and processed. The program detects and recognizes faces and based on the obtained labels, the database is updated and the processed photo is saved and returned to the front end web portal.

B. Use-Case

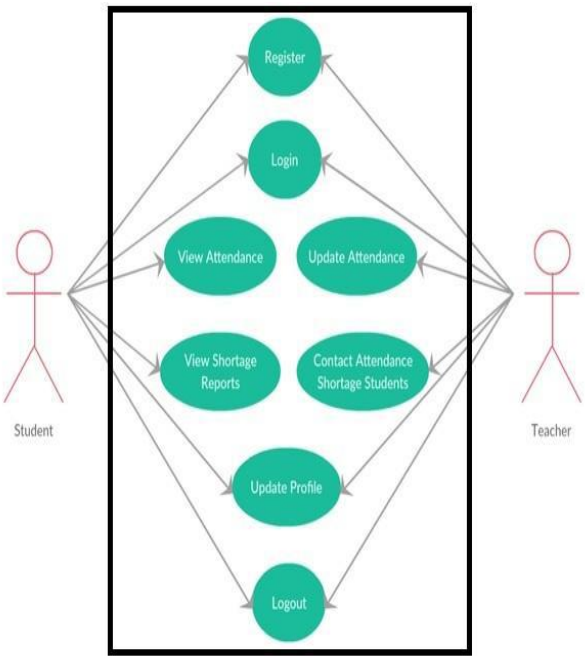

Fig4. Use Case Diagram

The use case diagram in Fig4 showcases the various features of the Face recognition attendance system. The web portal has functionality to handle logins and logouts, register and update new users as either teachers or students. Students have the ability to view attendance in each subject and view shortage reports. Teachers have the ability to view attendance for their respective subjects and contact students who have attendance shortage. Furthermore, teachers have the ability to leverage the Face Recognition Attendance System where they can log and update attendance by simply uploading a photograph of the class to the portal. Additionally, to cope with possible errors due to misclassifications, teachers have the ability to manually edit attendance values for each student individually.

VI. IMPLEMENTATION

A. Pre-Processing:

Images fed to the algorithm are grayscale as color has no effect nor contribution to the facial detection or recognition stages. OpenCV methods are used to convert all images to grayscale before any processing of that image occurs at any stage.

B. Face Detection:

We employ multiple classifiers to detect images. We use one classifier based on Local Binary Patterns Histogram and the other using Haar Cascade features. The faceExtractor method uses the classifier to extract and return face coordinates in an input image. This method is used in the data preparation or model training stage to train each face with its associate label. The faceExtractor method is called multiple times in the face recognition stage where it is used to detect and return multiple faces which are present in the final test image of the class.

C. Model Training:

The model is trained by passing 20-25 training images of the faces of each of the students. The training method data_prepare detects the face using the faceExtractor method and assigns this image to the particular unique student label. Training is reduced to a one time process thereby removing the overhead of having to retrain the model during every execution. This is achieved by the pickle module and cv2's built in object serialization. Environment variables are stored as byte files which can be loaded into other programs. The classifier's trained XML file is stored on the disk. This XML file can be loaded as is to a new instance of the classifier in another python program.

D. Face Recognition:

Firstly, the classifier and environment variables from the training program are loaded into the environment.

The testing program is invoked from the front end when the user presses the "Update Attendance" button and uploads the image. This is done with the help of the exec() command in PHP which invokes the command prompt. The testing program uses the faceExtractor image to first detect the faces in the test image. Then, one by one, a classification is done for each face. The method also creates and saves a copy of the test image (which is also displayed on the web portal) containing demarcated labels of faces recognized and labeled.

E. Database Updating:

The test function returns the processed image which is stored to the server for display on the portal and returns a list of the recognized faces. This list along with the command line parameter of the Subject name of the teacher who was logged in, are used to update the database. Pymysql module is used for database operations between python and MySQL. First the database connection is established. Then a cursor object is instantiated. Two main queries are run. The first query gets and updates the total number of classes conducted. The second query is run to update the attendance values. The second query is run for each recognized student. Finally, database changes are committed and the connection is closed.

VII. RESULTS

Fig4. shows the front end as it is viewed by the teacher/end user.

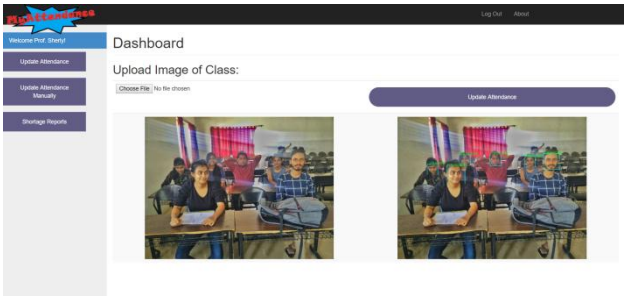

Fig4. Teacher Dashboard Update Attendance Page

The update attendance pane shows a file upload button where the teacher can upload the photograph of the class. Pressing the Update Attendance button invokes the PHP program which:

- Uploads the image to the server
- The image is saved and the python program is invoked by passing the command line command to the exec() method
- The uploaded image is processed by the python script. The faces are detected, separated and labels are classified. These labels are processed and used to update the database.

- The user is redirected back to the teacher homepage and upon re clicking the update attendance pane the user is able to view both the uploaded and processed image for verification. The teacher can view that the attendance has indeed been updated by navigating to the Update Attendance Manually pane.

VIII. CONCLUSION

Using Machine Learning, Python3, and a web stack of HTML, CSS, PHP, JavaScript, the Face Recognition Attendance System has been developed. All modules from the front end, the processing python script and the back end database are working perfectly. We have initialized the database for 8 students. . The portal provided the students with a way of easily viewing their attendance and gaining useful insight. The teachers were provided a way of digitally updating attendance without the hassle of paperwork to mark and recall current attendance status. This system can take over the existing methods followed in institutions and can turn out to be very efficient and with many advantages, providing ease to the teachers, students and education as a whole. The Face Recognition Attendance System can be developed further by implementing newer face recognition methods which in the future may be made more accurate. Uploading the image is still a process which can be made more efficient. For instance, uploading can be done over the internet by way of a connected mobile application. Furthermore, the database can also be migrated to the cloud

REFERENCES

1. Komal, Er. Rakesh Singh, "A Review Paper on Recent use of Local Binary Patterns and Its Variations for Face Recognition," *IJIR 2016*.
2. Paul Viola, Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", *Accepted Conference On Computer Vision and Pattern Recognition*, 2001.

AUTHORS PROFILE

Ayush Atul Hate is an aspiring engineer majoring in Computer Science and Engineering under VTU in Bangalore, India. He is passionate about upcoming technologies such a Machine Learning, Data Science and contributes frequently to research in his college's Center Of Excellence for Big Data and Machine Learning. He strives to find, research and implement unique applications for theoretical concepts and bring the learning out of classrooms and textbooks into the real world to help individuals and the society.