Employee Monitoring using Image Processing

Tejaswini Dhupad
Information Technology
AISSMS IOIT
Pune,India
e-mail:tejdhuppad@gmail.com

Shubhada Waghmare
Information Technology
AISSMS IOIT
Pune,India
e-mail:shubhadawaghmare95@gmail.com

Nikita Mandhan
Information Technology
AISSMS IOIT
Pune,India
e-mail:nikitamandhan22@gmail.com

Monal Chokshi
Information Technology
AISSMS IOIT
Pune,India
e-mail:monalchokshi373@gmail.com

***Abstract*—**Now-a-days demand of Employee Monitoring System is increasing with a rapid speed. Along with check-in and check-out details, it is also essential to calculate the total working hours of employees. Existing systems are only focused towards check-in and check-out. Our proposed system will be the potential solution for this problem, after sign in to the web portal the media device will implicitly monitor the employee. Moreover the proposed system will also detect the drowsiness based on eyes blink rate. Face and eyeball detection will be implemented using Haar cascade and Contour detection algorithms respectively. Finally the monitored data will be uploaded on real time cloud for the access and necessary actions by remotely located HR against the employees. This will be definitely helpful in increasing the productivity of the organization.

Keywords-*Blink-rate, Drowsy, Employee, Face detection, Monitoring.*

*****

I. INTRODUCTION

Existing Labor contract management system requires manual integration of data and management of different processes. Company requires different systems for Human Resource Information management, Project management, Client & Product management. Since searching and tracking of information of employees become complex and time consuming. In this system there will be two roles that is Admin and Employee. Company HR will be Admin. He/ She can calculate salary of employees and can set performance star for employees. In Employee side :A camera can detect if employee is present or not in front of computer and time is stored in database for which employee was absent. A Camera can detect employee eye if he closes his eyes then time is stored in database for which eye was closed to calculate drowsiness time.

Maintaining the attendance is very important in all the industries for checking the performance of employee. Every industry has its own method in this regard. Some are taking attendance manually using the old paper or file based approach and some have adopted methods of automatic attendance using some biometric techniques. This system uses the face recognition approach for the automatic attendance of employee in the classroom environment without employee's intervention. This attendance is recorded by using a camera attached in the classroom that is continuously capturing images of employee's, detect the eyes in images and compare the detected eyes with the database and mark the attendance. Camera takes the images continuously to detect and recognize all the employee in the classroom.

II. DROWSINESS DETECTION ALGORITHM BASD IN IMAGE PROCESSING

In this section, we focus on improving the detection rate of drowsiness of an employee by designing and then optimizing an algorithm based on Image processing and Machine learning. For us, only one sensor, the front camera on laptop, is needed, since we merely analyze employee eyes' open/closed states, the most distinguishing feature while fatigue

A.The Drowsiness definition for Laptop

It is mentioned that when people are fatigue, percentage of closure of eyelids (a.k.a PERCLOS) will exceed 32%, the time of continuous closing eyes will exceed 2.2 seconds and it is said that the frequency of wink will raise 64% when people are fatigue.

B. Haar Cascade Algorithm

It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, Haar features shown in below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

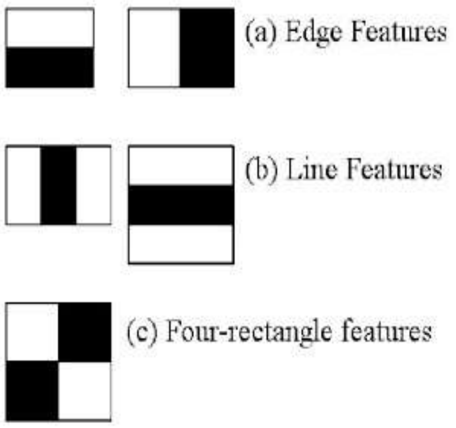

Figure 1. Haar Cascade Features

But among all these features we calculated, most of them are irrelevant. For example, consider the image below. Top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applying on cheeks or any other place is irrelevant. So how do we select the best features out of 160000+ features? It is achieved by Adaboost.

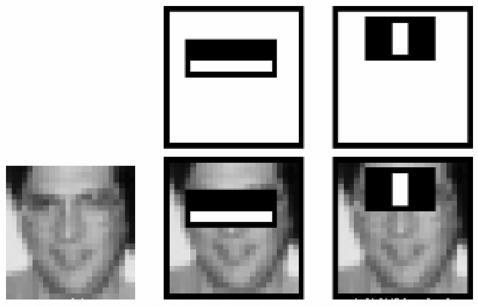

Figure 2. Haar Cascade filtered features

For this, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. But obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that best classifies the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then again same process is done. New error rates are calculated. Also new weights. The process is continued until required accuracy or error rate is achieved or required number of features are found).

C. Contour Detection Algorithm

Early approaches to contour detection aim at quantifying the presence of a boundary at a given image location through local measurements . The Canny detector [3] also models edges as sharp discontinuities in the brightness channel, adding non-maximum suppression and hysteresis thresholding steps. A richer description can be obtained by considering the response of the image to a family of filters of different scales and orientations. Lindeberg [4] proposes a filter-based method with an automatic scale selection mechanism. More recent local approaches take into account color and texture information and make use of learning techniques for cue combination. Martin et al.define gradient operators for brightness, color, and texture channels, and use them as input to a logistic regression classifier for predicting edge strength. Rather than rely on such hand-crafted features, Dollar et al. [5] propose a Boosted Edge Learning (BEL) algorithm which attempts to learn an edge classifier in the form of a probabilistic boosting tree [6] from thousands of simple features computed on image patches.

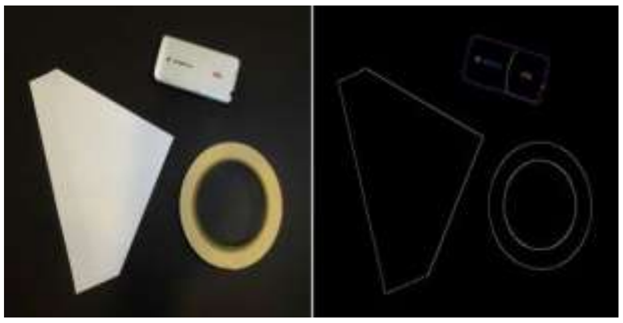

Figure 3. Contour Detection

III. OPTIMIZATION OF DETECTION ALGORITHM

The original algorithm model is easy to understand. However, it doesn't work on real world effectively. So we optimized the algorithm specifically for every problem in practice. Solutions and optimizations are listed below.

A. Find the Optimal Parameters Combination by Enumeration

We have four choices about input layer size (Figure 4.), and four choices of number of hidden layer nodes: 9, 16, 25, and 36.

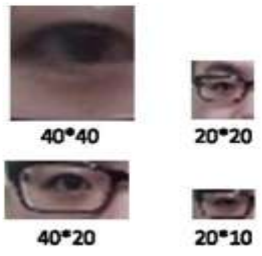

Figure 4. Four choices of input layer size.

We enumerate every combination of parameters, do experiments on multi-user training set, and then we find the best parameters combination according to the experiments. Due to space limitations, we cannot show the statistics of all experiments here.

According to results of each penalty coefficient, we can conclude that (1)The 40 × 20 input layer size is the best; (2)Increasing number of hidden layer nodes has little influence to the accuracy, since the lines are quite horizontal.

It is showed in the results of each number of hidden layer nodes that (1)The 40×20 input layer size is the best; (2)Overall, we get the highest accuracy at Penalty Coefficient λ = 0.3.

Taking the factors into account, we could safely arrive at the conclusion that the best combination of parameters is (1) input layer size = 40×20; (2) number of hidden layer nodes = 16; (3) penalty coefficient λ = 0.3.

We also chose 40×20 as model size because a rectangle(not square) size model will includes less redundant information like the spectacle frame and eyebrow comparing to 40×40 and 20 ×20, and it is the most natural size, which means less zooming, regarding distance between smart phones and driver comparing to 20×10[1].

B.*Image Pretreatment*

We actually can get position of eyes by Android API FaceDetector.Face [10] easily. To increase consistency of eye images, we do not just cut 40×20 pixels of the eye, instead, images are zoomed to 40 × 20 after cutting by size. $\text{eyeDistance} \times \frac{\text{eyeDistance}}{2}$ as shown in Figure 5.

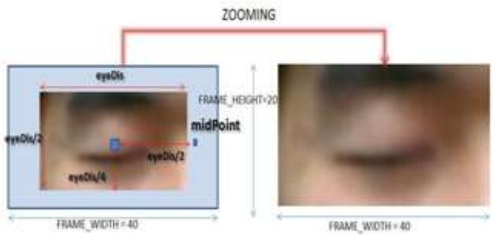

Figure 5. Eye image is zoomed to 40×20 from size. *EyesDistance* and *midPoint* are obtained from Android API Face Detector.

Before input layer, we need to do image pretreatment, including de-noising by decreasing images' brightness difference and contrast difference for better overall consistency, since less redundant information means more accuracy. The effects of image pretreatment are showed in Figure 6.

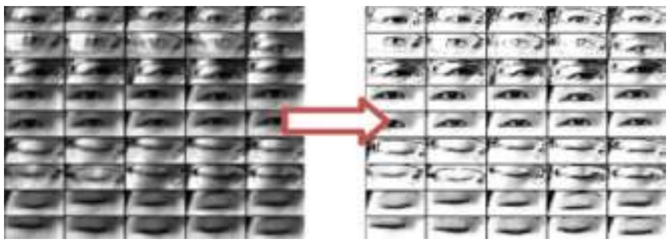

Figure 6.The results of image pretreatment.

IV. EMPLOYEE MONITORING ARCHITECTURE

The following figure 7. will explain the employee monitoring process:

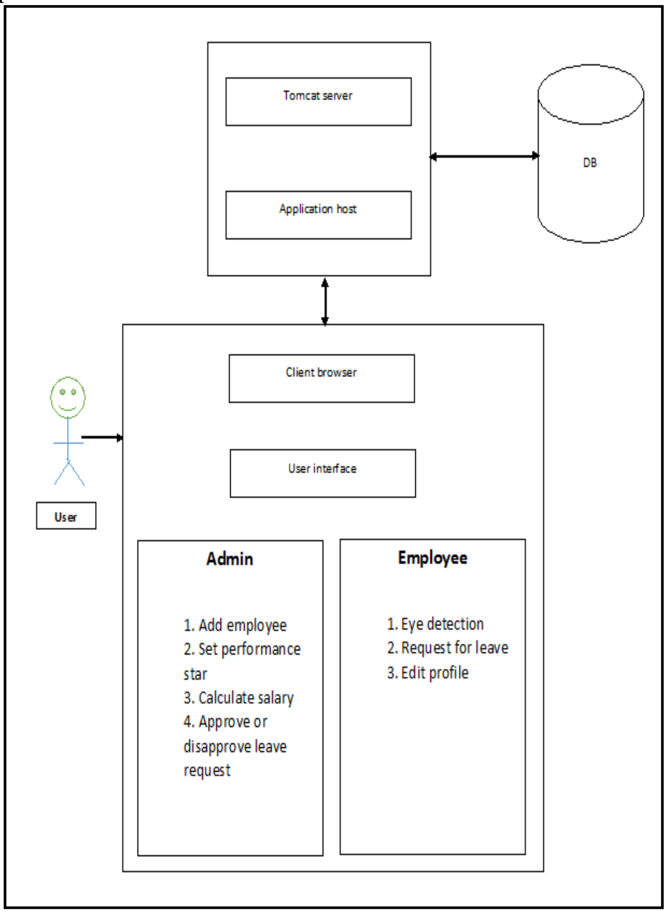

Figure 7. System Architecture

In this fig g. the employee would log into the company's website. As soon as the employee logs in the media devices starts monitoring the employee. The proposed system would also detect the drowsiness of the employee based on his blink rate. This monitoring would use two algorithms namely Haar Cascade and Contour Detection explained above. Finally the monitored data would be uploaded to the database and would get forwarded to the Human Resource (HR) of the company. Then finally, the salary of the employee would be calculated as well as the performance star of the employee would be calculated according to the calculated working period.

V. CONCLUSION

This paper presented the motivation, design, optimization and evaluation of a drowsiness detection algorithm based on Image Processing. It is implemented on laptop using only the front camera. The algorithm only focus on employee's eyes, obtained by Desktop API, classified by Haar Cascade and Contour, and then used to discrete-approximate several indicators including PERCLOS, blink time and blink rate to evaluate the fatigue level of the employee.

REFERENCES

[1] Sober-Drive: A Smartphone-assisted Drowsy Driving Detection System- Lunbo Xu, Shunyang Li, Kaigui Bian, Tong Zhao, Wei Yan Institute of Network Computing and Information Systems School of EECS, Peking ULunbo Xu, Shunyang Li, Kaigui Bian, Tong Zhao, Wei Yan Institute of Network Computing and Information Systems School of EECS, Peking University Beijing, China 100871 University Beijing, China 100871

[2] Paul Viola and Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001.

[3] [22] J. Canny, "A computational approach to edge detection," PAMI, 1986.

[4] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," IJCV, 1998.

[5] P. Dollar, Z. Tu, and S. Belongie, "Supervised learning of edges and object boundaries," CVPR, 2006.

[6] Z. Tu, "Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering," ICCV, 2005