Touch-less Face Enabled Time-Clock

Vaibhav Chhajed 50338182, Rugved Hattekar 50320920 and Nanditha Nandanavanam 50318504*

I. Introduction

The traditional method of keeping a record of work hours included manually keeping a list with timed entries which is often time consuming. The solution to this was using ID cards to swipe in or swipe out which provided automation to this process. However, there was still an issue of individuals giving proxy entries on behalf of the other individuals. Hence, there is a need to come up with an approach to make the time tracking more system accurate, efficient and authentic.

The project aims to develop with a system that can efficiently track the work hours of individuals in an organisational setup by keeping a record of their individual clock-ins and clock-outs on a daily basis. Further, the Real time detection of individuals and matching them against the previously enrolled individuals in the database and System-User interaction with voice commands adds the aspect of bio-metric authenticity to the system.

II. System Components

A. Graphical User Interface

This module is the front end of our system that monitors and captures responses from the user as well responds to them based on the functionality. The Tkinter module is a open source standard library which interfaces with the Graphical User Interface toolkit of Python. This module provides a User Interface window that can be partitioned into several frames. The frames can further be used to create a canvas for displaying live feed from webcam or a chat box that displays the voice commands sent by the user or even simple labels with information for the user. In this project, this module was used to create a window that shows the conversation between the system and the user using a scroll-able chat box. The system displays instructions to user like choice of voice commands as well as notifications like face recognised or unknown person in the chat box. Also, the voice commands spoken by the user are captured and displayed here so that the user can verify that their voice command was correctly captured or not.

Fig. 1 Main Window of the GUI

B. Voice recognition

Speech recognition involves receiving speech through the device microphone and checking the speech recognition service against a list of known vocabulary to recognise a word or a phrase. This module creates a Recogniser instance and uses the Google Web Speech API method for recognising speech from an audio source and then converts it to a text. This is used for recognising voice commands from the users.

*Department of Computer Science and Engineering, University at Buffalo.

C. Face Detection and Recognition

The first step in the face recognition is face detection. The detection module uses OpenCV to create object of class VideoCapture that opens webcam connected to the computer at use and reads frames from webcam in infinite loop into a display from in the GUI. For face detection we need a face, which in this project is captured when the new user is trying to register himself/herself in the system. At the time of registration the user stands infront of the camera, once the image is captured, he's asked to show an UBID Card, from which the unique UBID number is extracted and stored with the face image in the data set. Once this image is detected the HOG features are extracted, if the HOG features of extracted images matches with data set of general faces HOG features, the face is detected and the stored in the data set. This HOG Features will later help us in a to recognise any unknown image. Using this technique we keep adding more faces to our local data set. The next step is face encoding, for this we use a simplest approach to directly compare the unknown face we found with all the pictures we have of people that have already been tagged. When we find a previously tagged face that looks very similar to our unknown face, it must be the same person.

D. Optical Character Recognition

This module uses Google Cloud Vision API that performs feature detection on the scanned image to recognise text. The Google Cloud Vision API takes complex machine learning models centered around image recognition and formats it in a simple REST API interface. It encompasses a broad selection of tools to extract contextual data on images with a single API request.This is used to extract the credential information of the individual by scanning their ID card in the webcam. The recognised text is then dumped to a json file.

E. Logging entries

This module handles the logging of the individual's clock in and clock out entries with the timestamp on a daily basis and maintains a record of the information. This module further allows querying the data based on individuals on a daily/weekly basis. This module also keeps a record of exceptional entries like two consecutive clock ins or clock outs.

III. System Architecture

The Touch-less Face enabled Time Clock System comprises of several functional modules that interact with each other. The foremost component is the front end of the system is the Graphical User Interface that monitors and captures responses from the user as well responds to them based on them. It consists of a Tkinter package based Window that is partitioned into frames for displaying live feed from webcam and for displaying the voice commands given by the user. The webcam is used for real time capturing of the individual trying to clock in. This input is then passed to the Face detection module that detects the face from the image and passes it to the Face recognition system to match the image with those that are already stored in the database using HOG features. In the scenario, where there is no match found, the GUI displays this and asks permission to enter their credentials into the database. The user is then requested to show their ID card via webcam. The control then goes to the Optical Character recognition module that scans the ID card and extracts the credentials in the form of the text. This information is then added to the database. The system then logs the information of clock in or clock out of the individual with timestamp.

IV. Project execution

The project can be run by the "python test.py" command.
The following packages need to be installed:
1.Face recognition
2.Google Speech Recognition
3.PyAudio
4.Google OCR
5.Pillow
6.Spacy
7.OpenCV
8.Tkinter

The requirements.txt file in the project can be referred for further details.

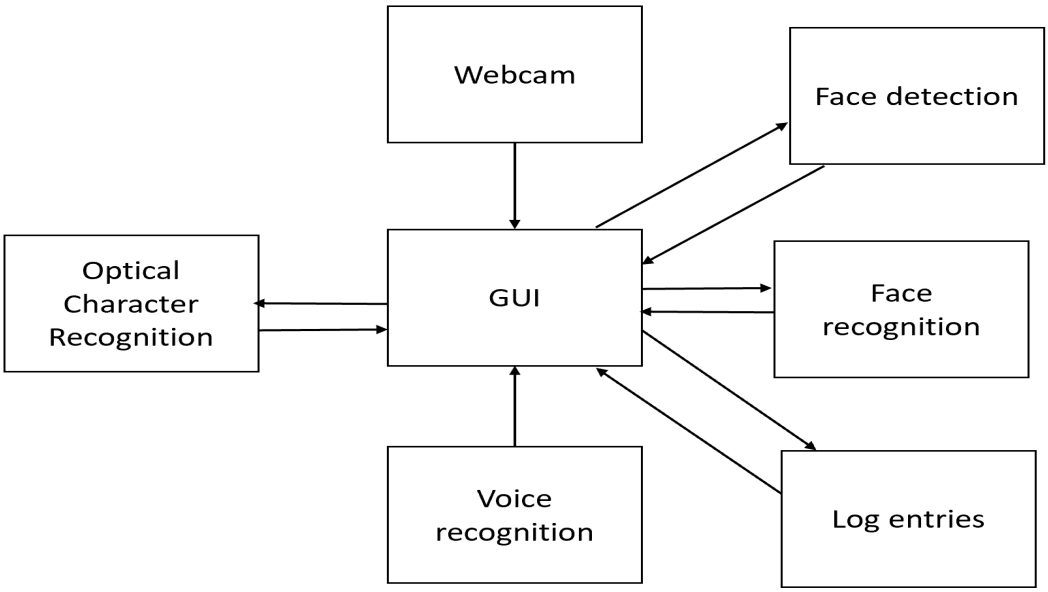

Fig. 2 Touch-less Face enabled Time Clock System Architecture

V. Major Challenges Overcome

In this project we are focusing on creating a vision based complete software package, which will be ready to implement at any facility for face detection enabled clock-in and clock-out. In this project we are dealing with face recognition, face detection and voice operated touch less system. This opens up to a combination of tasks altogether. The major challenge was of face recognition and detection, as in this problem we need to store the user image and use it every time the same user tries to log in or log out, and maintain a log file with all the clock-in and clock-out details. This require that the image name should be same as person name with some unique numeric key so that the two users with same name can log in anytime. For this we used UB Card and extracted data from UB card like name of user and person. To extract this we used spacy a NLP Library.

VI. Conclusion

This project was able to come up with a vision based approach to make the task of recording clock ins and clock outs of individuals in an organisation on a daily basis less time consuming by minimising manual work, more accurate and efficient, more secure by using face recognition to verify every entry as well as more contemporary by adapting voice based touch less system.

VII. References

[1]https://realpython.com/python-speech-recognition
[2]https://realpython.com/setting-up-a-simple-ocr-server
[3]https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78
[4]https://wiki.python.org/moin/TkInter
[5]https://www.pyimagesearch.com/2014/11/10/histogram-oriented-gradients-object-detection