Face Recognition Attendance System using Microsoft Azure Report

TABLE OF CONTENTS

ABSTRACT

In this proposed system, the system is initiated by the pc .After it triggers then the system starts processing the image for which we want to mark the attendance. Image Capturing phase is one in which we capture the image. This is basic phase from which we start initialising our system. We capture an image from a camera which is checked for certain constraints like lightning, spacing, density, facial expressions. The captured image is resolute for our requirements. Once it is resolute we make sure it is either in .png or .jpg format else it is converted. We take individuals different frontal postures so that the accuracy can be attained to the maximum extent. This is the training database in which every individual has been classified based on labels. For the captured image, from an every object we detect only frontal faces from Microsoft Azure face recognition API which detects only the frontal face posture of an every individual from the captured image. This detects only faces and removes every other parts since we are exploring the features of only faces. These detected faces are stored in the test database for further enquiry. Features are extracted in this extraction phase. The detected bounding boxes are further queried to look for features extraction and the extracted features are stored in matrix. For every detected phase this feature extraction is done. Features we look here are Shape, Edge, Colour and LBP. Face is recognised once we completed extracting features. The feature which is already trained with every individual is compared with the detected faces feature and if both features match then it is recognised. Once, it recognises it is going to update in the student attendance database.

INTRODUCTION

PURPOSE

The purpose of this report is to follow up a project on face recognition attendance system and give insight on how feasible it is to use a face recognition attendance system in a university environment.

SCOPE

The system should be built to be used for a prolonged period of time anywhere in the university campus where attendance would be tracked.

METHOD

Building such a system from scratch using the Python programming language helped achieve a better understanding of the field as well as its advantages and disadvantages compared to other biometric authentication methods. After some research, the decision to do face using an OpenCV library for Python and face recognition using Microsoft's Face API was unavoidable due to not having a system that could reliably do recognition inthe project's circumstances.

FACE RECOGNITION WITH MICROSOFT'S FACE API

The recognition part of the project was made using Microsoft's Face API. The reason is that Microsoft's API offers the ability to create, delete, and update a face list, which represents a group of pictures that must have only one face in them, that can be used to compare a face from outside the list against all the faces in the list and find a match. Therefore, these face lists can be used to their full potential in such an environment.

API LIMITATIONS

However, there are a few restrictions when using Microsoft's Face API: a free account can make 30.000 calls to the API per month and 20 per minute, whereas paid accounts can make 10 calls per second; only 64 face lists are allowed in one subscription; a face list cannot have more than 1000 faces; once a face has been added to a face list, the user receives the ID that was associated to that face, but there is no way to physically see what face/picture is represented by that ID anymore.

PROBLEMS ENCOUNTERED AND THEIR SOLUTIONS

Not being able to retrieve a face after adding it to a face list

This issue makes it difficult to find mistakes in the system, such as a student ID being linked to the wrong face, which can greatly reduce efficiency and accuracy, especially in an environment where such a system would be used on a great number of students on a daily basis. It would be impossible to know when a mistake has been made without being able to check if a student ID matches the right face and therefore a solution is essential for the system to be used appropriately.

SOLUTION: A work around to this issue is to keep track of all the face lists by creating a folder for each new list. Each time a face is added to a face list the same picture is copied to the face

list's folder with the name of the student ID it is associated with. It can be a naïve solution if implemented to work on a local machine since if the application would be used on a different machine, it would not be able to reproduce all the folders and pictures that were created on the previous machine. However, if the application would be connected to the university's servers and the solution would create the folders on the mentioned servers, then it would be easy to keep track of all the face lists regardless of what machine is being used.

OTHER POTENTIAL SOLUTIONS:

- A solution that avoids the use of folders would be to upload the pictures in an online database, but the downside is that storing a massive number of pictures in a database negatively impacts its performance and maintainability.
- A different solution is to upload the pictures on a photo-management website or on cloud storage. The main disadvantage of this approach would be uploading confidential information and/or pictures of the students on a website that is not guaranteed to be secure.

Having a limited number of calls to the API

It is a minor but inconvenient issue for the prototype as the calls to the API could be reduced by slowing down the application, however a final product would need to make more calls than the limit for a free account. As mentioned above, a free account can make 30.000 calls in a month and only 20 calls a minute, which means that the application cannot run for longer than a few seconds since it can only make a few calls even before it starts the face recognition system. The first call, to list all the existing face lists on the account, is made when the user goes to the face list menu; creating, deleting, updating a face list, adding a face to a face list and deleting a face from a face list would all make one call to the API; after the webcam is started, each snapshot has to go through Microsoft's face detection to obtain the face's ID, which is then passed to the face recognition API, returning the face ID of the recognised face, then makes another call to the API to get the student ID that is matched to the face, therefore each snapshot makes in total 3 calls. For the application to run continuously for a longer amount of time, in the best case scenario where only one API call is made before starting the face recognition system, a snapshot of the detected faces would be saved every 10 seconds. It is enough for testing purposes, however such a pace in a university environment would be very slow and inefficient.

SOLUTION: The only solution for this issue would be to use a paid Microsoft account which allows up to 10 calls per second. This would significantly improve the speed and consistency of the entire process.

CONCLUSION

A face recognition system using Microsoft azure would certainly speed up the process of checking student attendance in comparison to other biometrics authentication methods and in the right circumstances it would be able to match their accuracy. Nowadays there are a wide variety of software, whether it is a Face API like Microsoft's or a library like OpenCV, that makes face recognition accessible and reliable and is constantly improving. Each software imposes various restrictions, such as the limited number of calls you can make to Microsoft's Face API. However, using more than one software can reduce these restrictions and lead to better results.

RECOMMENDATIONS

If this system would be implemented in a university, the following would be recommended:

- The camera that would be used for the face recognition should be placed in front of the lecture theatre door at a distance of 3 feet and a height of 65 inches. This would assure a better accuracy from the face recognition system.
- The pictures that are added to face lists should ideally be the same as the pictures used for student IDs since they are guaranteed to contain only one face.
- The application should be hosted on the university's servers to ensure consistency with the folders corresponding to each face list created and with the pictures in each folder corresponding to what faces each face list contains.

APPENDIX I. EVALUATION MATRIX SCORES

Area	Scoring System	Score	Reason
Maturity	1 = Idea 5 = Mainstream Product	3	Some companies already have such systems in place to check attendance and there are a few products on the market that can be used for this purpose.
Technology (Adoption timescales)	1 = > 3 years 5 = < 3 months	4	Any type of a video camera would be compatible with a facial detection and recognition system. The software would run on any operating system.
Business Process (Adoption timescales)	1 = > 3 years 5 = < 3 months	4	The cameras and machines cost would have to be taken into consideration.
Adoption Overview	1 = v long time 5 = very short	4	Placing the cameras where they are needed and installing the software on a machine is a quick process.
Existing Technology (Impact)	1 = v large impact 5 = very little	4	There is little or no negative impact on existing technology.
Resources Required	1 = v large impact 5 = very little	3	Depending on the system used the requirements could differ, such as a system could only work on machines that have Java installed.

Scope	1=very difficult 5=very easy	4	It could be used to check attendance in the workplace or at school and the software would run on newer machines as well as older ones.
Usability	1=very difficult 5=very easy	4	It is an automated system that can be straightforward to use, however staff would need to be trained in case the system malfunctions.
Security	1 = very poor 5 = excellent	4	It is be secure as it would detect people that are not recognised, however staff members would have to intervene and deal with the intruder.
Innovation Value	1 = low innov. 5 = high innov.	4	Provides an easy and quick way to check attendance at school or work.
Cost Effectiveness	1=very expensive 5=very cost effective	3	Camera will need to be installed in each lecture room and compatible software needs to be set up. There are commercial off-the-shelf products to buy, although the in-house development may be another possible solution.
Adoption Readiness Score	*<20 - not ready* *20-29 - emerging* *30-39 - Adoptable* *>39 Fully Ready*	*31*	*Facial detection and recognition systems that check attendance have been on the market for a while. It is not a new idea, but it is an efficient system even with today's technologies. Such systems are reliable in terms of cost and time and they are easy to implement.*
***Note**: Rows that have no highlight colour indicate the score value is not added to the adoption readiness total. Instead, the overview score for that area is used as part of the total score.*			