Global Journal of Engineering and Technology Advances
eISSN: 2582-5003
Cross Ref DOI: 10.30574/gjeta
Journal homepage: https://gjeta.com/

(RESEARCH ARTICLE)

Check for updates

A deep learning-based face recognition attendance system

Ubong David Essien * and Godwin Okon Ansa

Department of Computer Science, Faculty of Physical Sciences, Akwa Ibom State University Ikot Akpaden, Mkpat Enin.

Global Journal of Engineering and Technology Advances, 2023, 17(01), 009–022

Publication history: Received on 10 July 2023; revised on 29 September 2023; accepted on 02 October 2023

Article DOI: https://doi.org/10.30574/gjeta.2023.17.1.0165

Abstract

Deep learning-based face recognition system have produced high accuracy and better performance when compared to other methods of face recognition like the eigen faces. Modern face recognition systems consist of different phases such as face detection, face alignment, feature extraction, face representation and face recognition. This paper proposes a deep learning approach in developing a face recognition-based class attendance system. The Multitask Convolutional Neural Network (MTCNN) is used for the face detection and alignment phase and a lightweight hybrid high performance Deepface Python framework based on the 'Deepface' Deep Convolutional Neural Network is employed for the feature extraction, face representation and face recognition phases with FaceNet-512 pretrained model. Because Convolutional Neural Networks (CNNs) perform better with larger datasets, image augmentation will be used on the original photos to enlarge the tiny dataset. The attendance record is stored in a MySQL database and accessed by an Application Programming Interface (API) developed using Hypertext Pre-processor (PHP) CodeIgniter framework. Cosine Similarity is used as the similarity metrics to compare the facial embeddings. A sliding camera system is deployed to aids the full coverage of the class participants irrespective of the size of the class. The test result show that all class participants were correctly identified and captured in the class attendance register generated.

Keywords: Deep Convolutional Neural Network; Application Programming Interface; Facial Embeddings; Image Augmentation

1. Introduction

According to [10], empirical evidences have shown that there is a significant correlation between students who have poor attendances and their academic performances. Attendance is the single strongest predictor of college grades, according to a meta-analysis [4]. Face recognition systems have quickly surpassed other forms of biometrics such as Fingerprints, Radio Frequency Identification etc. as they use a set of features distinct to one person [7] According to [2], the issue of low-class attendance in higher learning institutions has been and continues to be a major concern for educators and educational researchers worldwide.

The proposed system will be implemented in combination with a single sliding camera architecture [6]. The goal of this research is to improve the efficiency of the attendance system, eliminate impersonation, create secure access to attendance data, and save time that would otherwise be spent in the lecture. Existing facial recognition systems can be roughly classified into four types based on how the face is represented: -

As previously mentioned, appearance-based, which employs holistic texture features.

Model-based face recognition, which uses the shape and texture of the face as well as three-dimensional depth information; Template-based face recognition; and Neural network-based face recognition.

* Corresponding author: Ubong Essien

Because neural network-based approaches outperform statistical approaches in terms of accuracy, we presented a deep learning-based facial recognition method for the creation of the attendance system. Face detection and recognition will be accomplished using a Deep Convolutional Neural Network (DCNN) in this paper. [18] showed that DCNN outperforms other standard approaches.

The proposed face recognition system is separated into two sections: face detection/alignment and extraction and feature extraction/facial recognition.

This research will employ the Multi Task Convolutional Neural Network (MTCNN) for face detection and Deepface for face recognition. The libraries utilized in the development of the proposed system are listed in Table 1.

Table 1 Python Libraries Used for the Face Recognition System

Stage	Library	Developer	Source
Face Detection/Face Alignment	Python MTCNN library	Ipazc	https://github.com/ipazc/mtcnn
Feature Extraction/face Recognition	Python Deepface Framework	Serengil and Ozpinar	https://sefiks.com/2020/02/17/face-recognition-with-facebook-deepface-in-keras/

1.1. Motivation

Developing a system that is cost effective, scalable and provides secured access to student's class attendance records.

1.2. Related Literature

[14] used OpenCV in their work. Python was used to create the application. While Raspberry Pi serves as the application's central processing unit. The open CV and the camera are both installed on the Raspberry Pi. The face database is created by photographing the student standing in front of the camera several times. The face database is created using approximately 100 different images of each student. All of these images are saved, and a face database is created as a result. Face detection on a human face is accomplished by matching a set of different Haar-like features. In this work, the Local Binary Patterns Histogram (LBPH) algorithm was used for facial recognition.

[7] provided a system to capture the faces of individual students using the Haar-Cascade algorithm applied to videos. Use line and edge features to get different facial features. The Hair Cascade technique emphasizes the most important regions of the face for detection, the so-called regions of interest (ROI). Other areas of the face not involved in image processing are also processed and processed. When faces are detected, they are fetched and stored. The Local Binary Pattern Histogram (LBPH) algorithm computes histogram values for each image stored in the database and compares them with computed histogram values for images extracted from captured video feeds.

[5] presented a facial recognition attendance system based on deep learning. Face recognition is based on a comprehensive cascading multitasking framework that leverages the natural correlation between recognition and orientation to improve performance. This framework uses a cascading architecture with her carefully constructed 3-layer deep convolutional network to predict viewpoints and landmarks from coarse to fine. This model uses his three convolutional networks: P-Net, R-Net and O-Net. This facial recognition approach works well for non-frontal faces. For training he used two FaceNet models. Increasing the size of the dataset improved the detection.

[11] provide a system that includes facial recognition and recognition strategies. Video surveillance uses spatio-temporal frequency object mining (STFOM) to detect anomalous human behavior. Viola Jones Face Detector detects faces. This system is based on his three concepts. Integral Image, AdaBoost, Cascade Structure. After face recognition, features such as histogram of directional gradients (HOG) and weighted local binary patterns (WLBP) are extracted and used in Orthogonal Locality Preserving Projection (OLPP) for face recognition. Variation in detected face poses significantly reduces OLPP-based face detection. To solve this problem, the system uses a pose-invariant HFO OLLP-based face recognition algorithm that efficiently uses OLPP's HFO, HFD, HOG, and WLBP features.

[12] offer a face recognition system based on Principal Component Analysis (PCA). The framework outlines the whole face recognition process, with several versions accessible for varied requirements at each phase.

[3] provides a single-camera facial recognition system that combines directional gradient histograms for facial recognition with deep learning algorithms to generate and compare his 128-dimensional facial features. For recognized students with the correct subject ID, the system estimates attendance in real time. The system then creates and saves her Excel spreadsheet. The system was divided into two parts, the front-end side consists of a graphical user interface (GUI) that uses her Electron JS, which is a JavaScript stack, and acts as a client, while the back-end side is split between her two components. The backend consists of logic, written in Python, and acts as a server. Neither language can communicate directly with each other, so library-less interpersonal communication (IPC) techniques are used as a bridge for communication between these two languages. The Zero PC Library allows Electron JS to call Python functions and exchange data over the Transport Control Protocol (TCP).

The limitations found are stated below:

- Low level accuracy if there is any change in orientation or illumination as this system is only suitable for frontal orientation of images. The system uses a single camera architecture which is affected by camera blind spots and were not suitable for large room spaces. The attendance record is presented in excel sheets and entries can be modified. Lack of an authentication mechanism to protect the attendance record from unauthorized access and modification.
- The proposed system will employ a deep learning-based approach where the output is not affected by changes in illumination and image conditions. Our training data will consist of both side and frontal face images of students, this will improve the recognition for students if the side view is captured during the attendance marking process. To address the issue of unauthorized access to attendance record, our system will incorporate a secured API.

2. Methodology

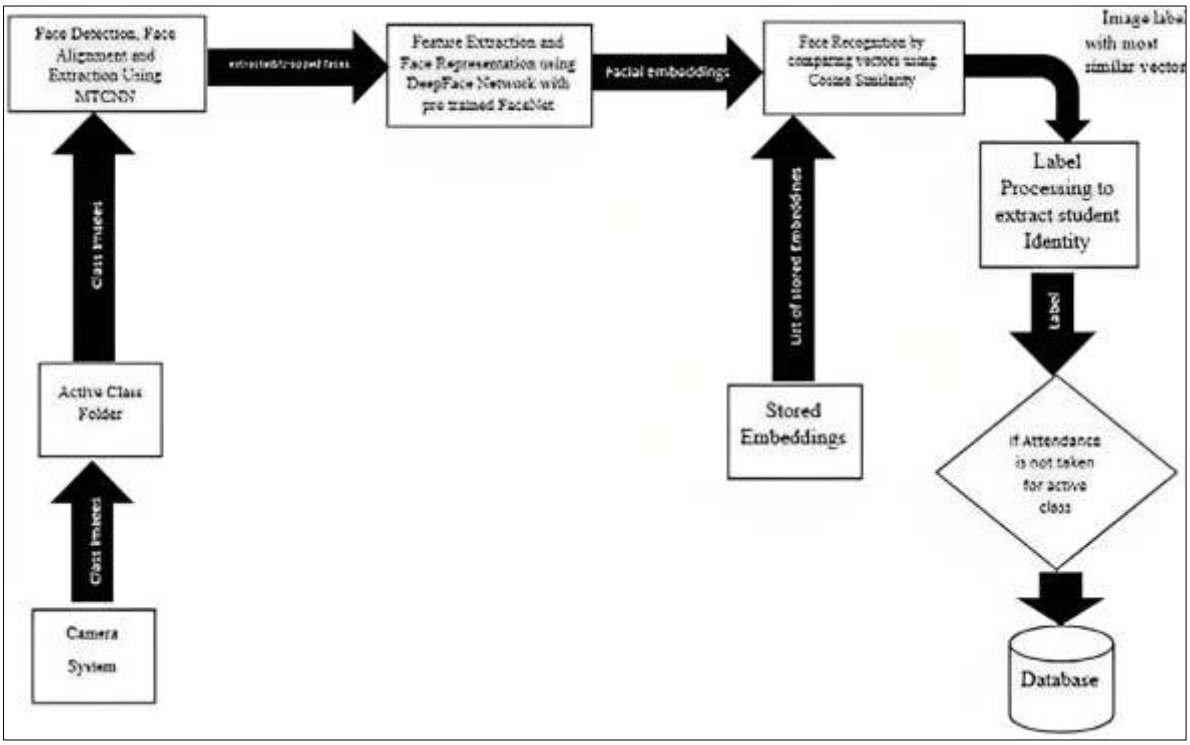


Figure 1 Architecture of the face recognition attendance system

The proposed system uses a sliding camera architecture proposed by [6]. This camera system allows for capturing of class images irrespective of the class size and the ability of the system to segment images for different classes makes it effective for capturing class images for the face detection phase of this face recognition system. The captured class images are passed to the face detection module where participants faces are detected and cropped. Each detected face is uniquely identified by using unique random numbers in their file names. The face detection is implemented using the Multitask Convolutional Neural Network (MTCNN) by [23] that allows us to perform both detection an alignment. The

Face recognition phase of this system is based on Deepface Neural Network [18] with pretrained FaceNet 512 model by [16]. The FaceNet 512 is used because of the better accuracy. Image augmentation is carried out to create more copies of the faces in the face database. The detected faces are saved into a detected faces folder and passed to the feature extraction module where the facial embeddings are generated and compared against the facial embeddings of the known faces in the face database. Attendance is marked when the most similar face is found and the face identity is stored in a MySQL database. Each class is unique and class attendance is taken based on the class. Access to this attendance record is via API which authenticates every user request.

2.1. Face Enrolment

Facial images of volunteer students are captured using a camera and saved in a folder. Three (3) facial images per student were captured for a sample size of thirty (30) students.

Figure 2 Captured facial image of a student

2.2. Dataset Preparation and Augmentation

All faces were cropped to 160px by 160px and stored in a single folder with the registration number as the label of the folder. We used the Imgaug library to carry out image augmentation. The Imgaug library allows different filters and image properties to be applied at random to different copies of an image. Gaussian blur, linear contrast, gaussian noise and rotation were applied to each image to create a total of 10 images per student.

2.3. Face Detection /Alignment and Face Extraction

This research will employ Deep Convolutional Neural Network known as the Multi Task Convolutional Neural Network (MTCNN) for its face detection phase.

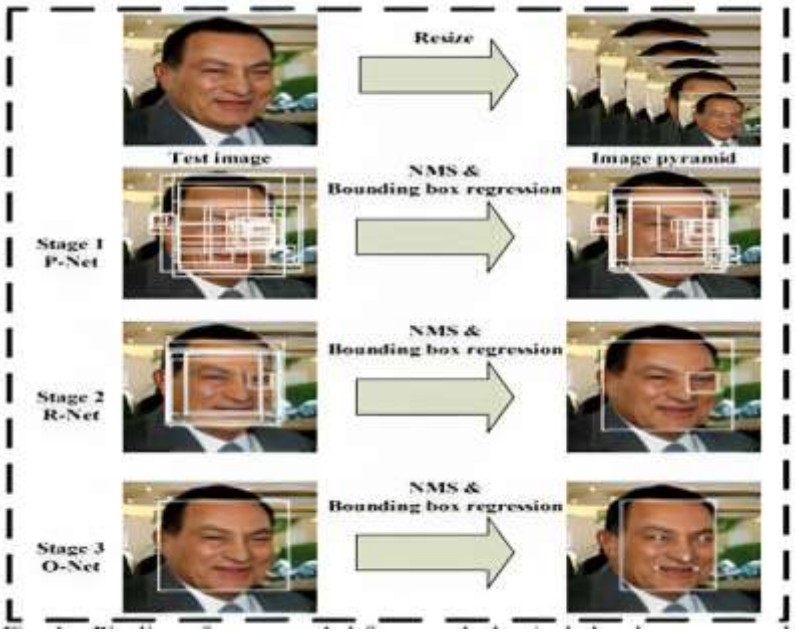


Figure 3 The MTCNN Pipeline [23]

MTCNN is a deep learning-based face recognition and face alignment method, which has a better face recognition effect and is more tolerant to changes in light, angle, and facial expressions in natural environments. At the same time, it has low memory usage and enables real-time face recognition [10]. MTCNN improves performance with face alignment and face recognition.

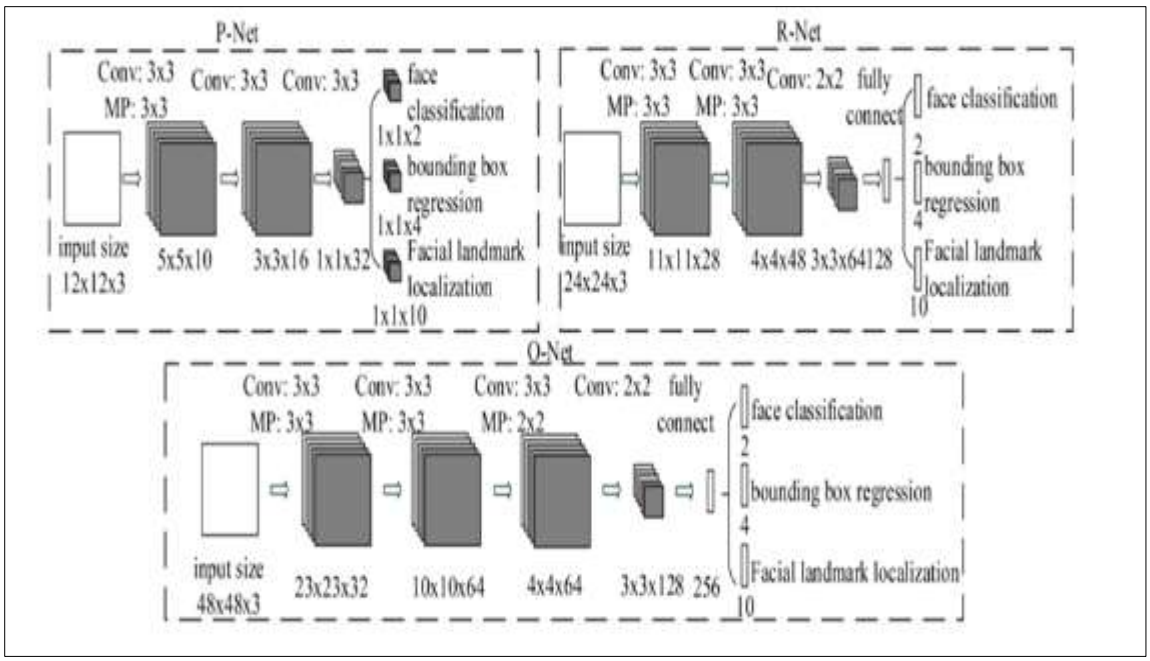
MTCNN consists of three neural network cascades. P-net, R-net, O-net. Candidate windows are quickly generated using a flat CNN. Then a more complicated CNN to adjust the window and reject many hidden windows. Finally, we use a more powerful CNN to improve our results and output the facial marker positions.

The MTCNN handles both the face detection and the face alignment for any input image. [23] enumerated the overall pipeline of the MTCNN is shown in Figure. 3.8. Given an image, the network first resizes it to different scales to create an image pyramid, which serves as the input to the three-stage cascaded framework. A summary of the operation of this network is as follows:

- Stage 1: A fully convolutional network dubbed Proposal Network (P-Net) was utilized to get the candidate windows and their bounding box regression vectors. The bounding box regression vectors calculated are then utilized to calibrate the candidates. Then, candidates with substantial overlap are merged using non-maximum suppression (NMS).
- Stage 2: All candidates are fed into another CNN called Refine Network (R-Net), which rejects a large number of false candidates, calibrates using bounding box regression, and merges NMS candidates
- Stage 3: In this stage, the aim is to describe the face in more details. In particular, the network will output five facial landmarks' positions.

Generally, The Network's task is to output three things: face/non-face classification, bounding box regression, and facial landmark localization like eyes, nose and corners of a mouth.

The Figure 4 depicts the MTCNN architecture as described by [23]



MP - Max pooling, Conv – Convolution

Figure 4 The Architectures of P-Net, R-Net, and O-Net [23]

An in-depth detail of the MTCNN Architecture is captured in the work of [10],[23]. The MTCNN is available to python via the TensorFlow.

2.4. Feature Extraction and Face Representation

According to [17], there are two common approaches to extract facial features: Geometric features and Appearance based methods. The feature extraction method employed is a feature-based method with different filters applied at different layers to extract low-level features, like simple edges and texture [18].

Features are parts or patterns of an object in an image that help to identify the object. Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set. The advantage of using deep learning is that we do not manually have to extract features from the image.

The feature-based or analytic method computes a set of geometrical face features such as eyes, mouth, and nose. The facial features vector is formed by combining the positions of various facial features and the outline of the face. The geometrical relationships are computed using the points' locations. Geometric features describe the shape and location of facial components, which are extracted to create a feature vector that represents the face [17]

Convolution Neural Networks performs the best in feature extraction with highest accuracy [9].

The FaceNet pretrained model used in this research is responsible for the feature extraction and feature representation of facial embeddings.

The most significant component of FaceNet, according to [15], is end-to-end learning. The triplet loss is used for face verification, identification, and clustering. FaceNet seeks to generate an embedding f(x) from an image x into a feature space Rd such that the squared distance between all faces of the same identity is modest regardless of imaging settings, however the squared distance between two face images from different identities is big.

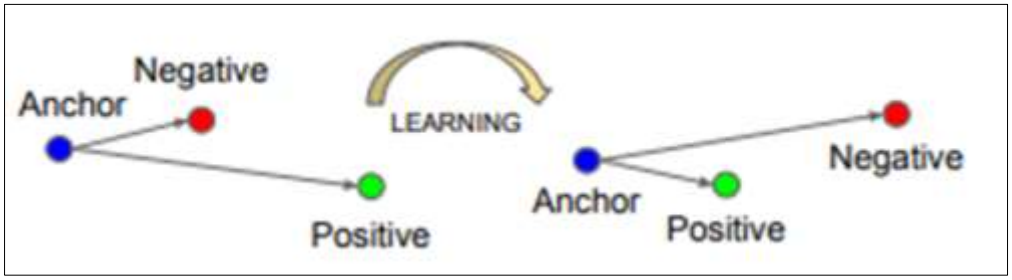


Figure 5 The Triplet Loss

Triplet loss, on the other hand, aims to create a buffer between each pair of a person's faces and every other face. This allows identity faces to coexist on the manifold while maintaining a distance, thereby distinguishing them from other identities.

In summary, triplet loss minimizes the distance between anchors and positives with the same identity and maximizes the distance between anchors and negatives with different identities. For more information on FaceNet, see the study "FaceNet: Integrated Embedding for Face Recognition and Clustering" [15]

Table 2 Models and their Input/Output Shapes [16]

Model	Input Shape	Output Shape
VGG-Face	224 x 224 x 3	2622
FaceNet	160 x 160 x 3	128
OpenFace	96 x 96 x 3	128
DeepFace	152 x 152 x 3	4096
DeepID2	55 x 47 x 3	160
Dlib	150 x 150 x 3	128

The primary purpose of feature extraction is to reduce machine training time and space complexity in order to reduce dimension [8]. The proposed system's feature extraction and face representation are based on the work of [18]. A Python framework was created by [16]. This Deepface framework encases some of the most advanced face recognition models available today, as illustrated in Table 2.

The proposed system uses the Facenet-512 pre trained model because it offers better [13]. This model outputs a 512-dimension vector. To reduce the cost of processing during the representation phase of the face recognition, Deepface extracts and stores representations beforehand. The algorithm for the generating the facial embeddings is given below:

- Step 1: Read the face database folder path
- Step 2: Check if folder path contains images, if true, create an extracted embeddings array.
- Step 3: For each of the images in the folder
  - Detect the face in the image using the MTCNN face detection and Extract faces from the image.
  - Extract facial features using the Deepface framework with the FaceNet-512 model from the image as embeddings
  - Push embeddings to extracted embeddings array created in Step 2 with image path as key and embeddings as value.
- Step 4: Serialize the extracted embedding object using the pickle library and save the file within the face database folder with representations_feature_extraction_method.pkl e.g representations_facenet512.pkl

2.5. Face Recognition

According to [19], face recognition methods include geometry-based methods, holistic methods, feature-based methods, hybrid methods, and deep learning methods.

A deep learning method is proposed in this research project. This is because models can be trained on vast amounts of data to create facial representations that can handle variations in the training data. This requires designing special features that are robust to different types of variations within the class (illumination, pose, facial expression, age, etc.) for each variation in the training data, as [19]. Therefore, CNNs can learn directly from training data. The deep learning face recognition network used in this study is the Deepface network with pre-trained FaceNet weights [1],[5],[16], [20], [21].

Deepface was created by the Artificial Intelligence Research Group at Facebook in 2015 according to the research publication titled “Deepface: Closing the Gap to Human-Level Performance in Face Verification” in 2014 [18].

The Deepface Architecture is an eight-layer Deep Neural Network (DNN) with an effective learning strategy that employs a very large labelled dataset of faces (137,774,071) to produce a face representation that generalizes well to other datasets [18]. It also has a powerful facial alignment algorithm based on explicit 3D face modelling. Its ability to produce near-human performance on labelled Faces in the Wild at 97.5 percent and a 50% reduction in error rates on the YouTube Faces Dataset makes it one of the most popular frameworks in computer vision [18]. DeepFace model expects facial image as input and represent it as 4096-dimensional vector.

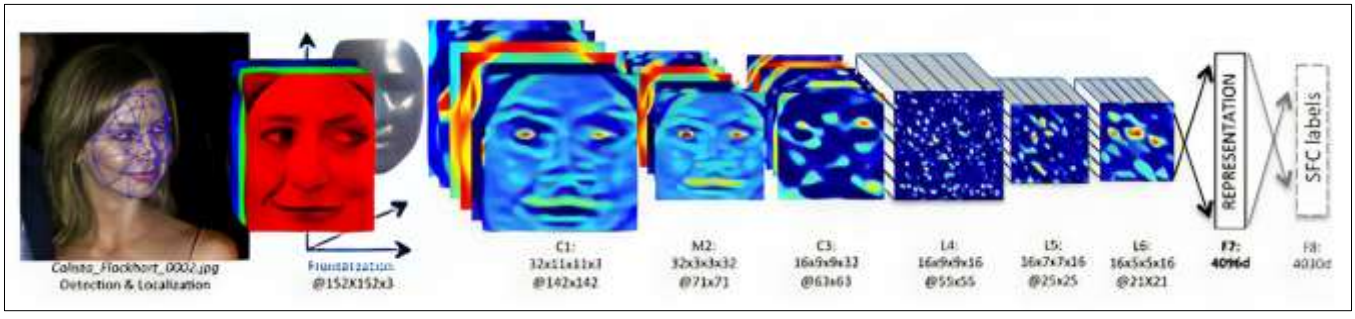


Figure 6 Outline of the DeepFace Architecture [18]

Deepface is available as a python framework developed by [16] for facial recognition and attributes analysis. This framework in python differs from the original work of [18] in the fact that it does not require the last F8 layer which is the output labels, instead the facial representation vector from the F7 layer is used and vector comparison is carried out.

One of the major advantages of using this framework is that DeepFace has the ability to accommodate different state of the art deep learning model. Deepface has some inbuilt methods that simplifies face recognition as described in the work of [1]. The Deepface face recognition python framework by serengil handles the face alignment,feature extraction,face representation as its core function.

Deepface uses verification methods to handle the face recognition process. This method compares the same-dimensional face vectors of two images passed to the Deepface Framework's validation module to find similarities. Similarity is an important consideration in classification and clustering tasks [15]. By default, the Deepface framework uses three similarity metrics to determine the similarity between two vectors: cosine similarity, Euclidean distance, and Euclidean L2. Finding faces in large datasets requires multiple rounds of face verification. The framework provides an out-of-the-box function, the Find method, for this task.

2.6. Class Creation

To create an attendance for a class, a class record must first be created in the database that contains information of the class. Every new created class is the active class for a given venue. This means that the last created class for a given venue is considered the active class for that venue. A venue must not host two class sessions. Every class has its own folder to store its class images and detected faces.

The information needed to create a class are:

- Class Title: The Title of the Class to be created
- The Instructor Name: The name of the instructor, lecturer or facilitator
- The Venue: The Venue of the lecture, which must have been pre-determined.
- The Class Date: The Date of the class.
- The Class Images Directory: The folder where the class images and processed images will be stored. This is created automatically by system.

Algorithm for creating a class: -

- Start.
- Input Class Name, Instructors Name, Venue, Class Date.
- Prepare the class folder label using the format Class Name_Class Date_random string.
- Make a directory with the prepared class folder label within the project image directory.
- Make a child directory inside the directory created in 3 with the prefix label “detected_” concatenated with the folder label of the parent folder.
- If both directories are created successfully
  - Insert class details to the database.
  - If Insert is successful.
  - Print success message.
  - else
  - Delete folder created in 4
- else
  - Go to 3
- End

2.7. Attendance Marking

The Deepface Find Method returns the file path of the image with the closest similarity. Similarity is a fundamental notion in the field of machine learning and pattern recognition [22]. Each enrolled student has a folder with the registration number as the label of the folder. This registration number is extracted from the label and stored in the MySQL database to mark the attendance for the student. The attendance is marked per class created.

2.8. Web Based Interface

Attendance records can be accessed by users (lecturers, management or parents). Each user must be authenticated before access to the service is granted. This web-based interface is created to communicate with the API to fetch and return the attendance record.

2.8.1. Implementation

In this paper, a web-based application is developed using Flask Python framework. The system is implemented in such a way that different classes can be hosted at various venues on the system.

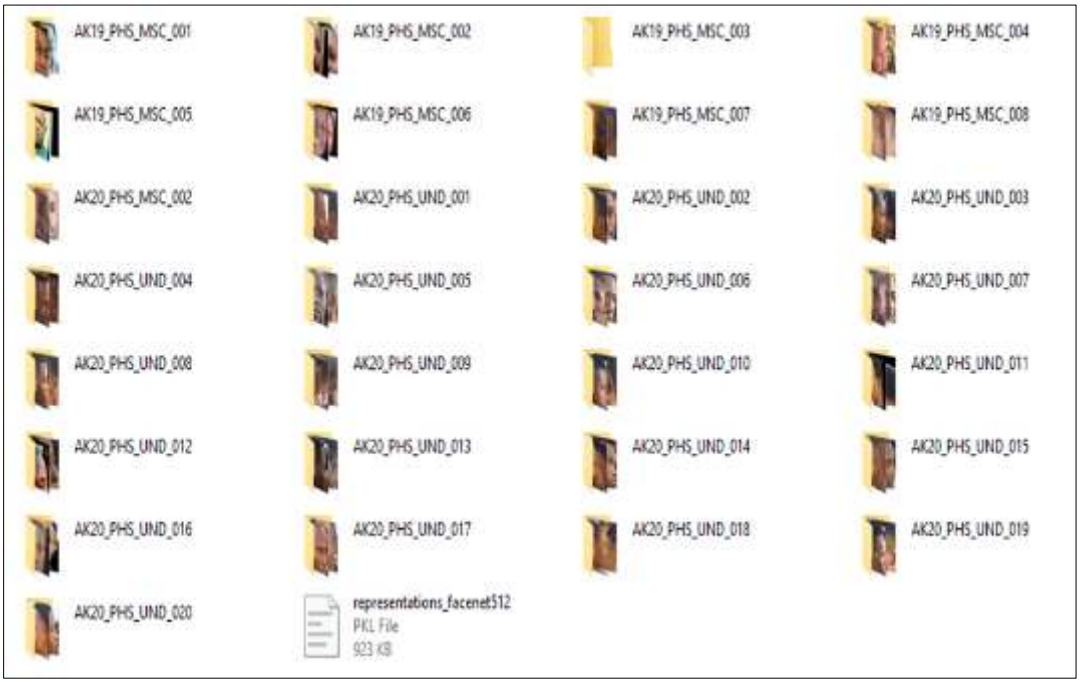


Figure 7 A Snapshot of face database

The facial embeddings of all the faces in the face database are generated stored in a serialized file in the format. This file will be used to recognize the faces during the similarity comparison stage. The format for the embeddings is given below:

embedding[‘img1_path’:[img1_facial_vector],‘img2_path’:[img2_facial _vector],…….. [‘imgn_path’:[imgn_facial_vector]]

To increase the number of images in the face database, data augmentation was done using the imgaug python library as shown in Figure 4.

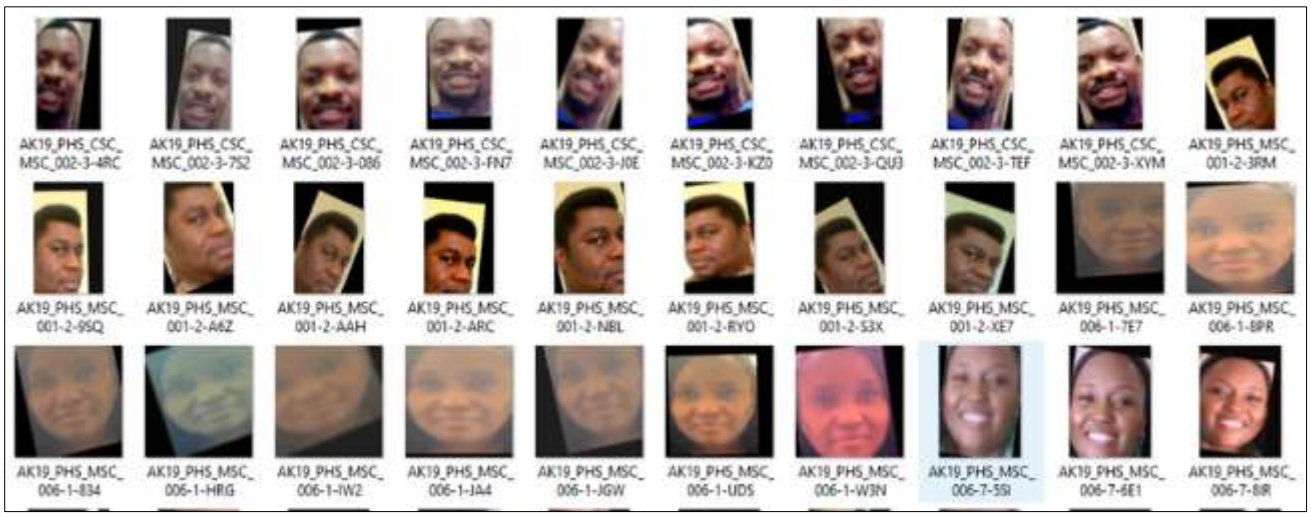


Figure 8 Image augmentation for database images

Figure 9 A sample class image

Figure 5 above was passed to the face detection module using the MTCNN and the result is shown in Figure 6.

Figure 10 Detected and cropped faces

These faces are stored within the detected folder for the current active class and are each passed to the face recognition and attendance marking modules for recognition and attendance marking respectively.

The face recognition system is implemented using the following: -

- Read the Active Created Class from the database and locate the folder to read the class images.
- Read all class image file in the Active-Class-Folder
- For each image in Active-Class-Folder detect and extract faces in the Images located in the Active-Class-Folder using MTCNN and save the extracted face image into the detected face folder within the active class folder with identifiers.
- Pass all the filenames of the detected faces in 3 as an array into the find method of the Deepface class.
- Extract the matched faces in the face database and retrieve their registration number from the file path.
- Insert the registration number returned if it does not exist into the attendance_tb table of the face_recognition_db database.

In this paper a web-based API developed using CodeIgniter is implemented for access to the attendance data, hence ensuring data security.

The attendance record can only be accessed by external users if they have been authenticated. The web-based API has an Authenticate method that checks three parameters assigned to each user. These parameters are: the username, the password and the secret key.

The username and password are chosen by the user while the secret key is autogenerated by the API and stored in the database. The API can be accessed either with a GET or POST method to get access to the attendance record. This is important in the verification of attendees at conferences by sponsors.

```
public function Authenticate(){
    $user=$this->input->post('user',TRUE);
    $pass=$this->input->post('pass',TRUE);
    $sk=$this->input->post('secret_key',TRUE);
    $count_dt=$this->api_m->Authenticate($user,$pass,$sk);
    if($count_dt > 0){
      // if the details are valid
      $_SESSION['auth_user']=$user;
          echo "<p style='color:green;font-family:arial;font-weight:bold'>Authentication is Successfull.</p><a
href='".base_url('dashboard')."' class='btn btn-success'>Continue</a>";
    }else{
      echo "<p style='color:red;font-family:arial;font-weight:bold'>Authentication Failed, Wrong Credentials!.</p>";
    }
  }
```

Figure 11 Snippet of the Authenticate Method of the API Class

```
public function get(){
    if(!isset($_SESSION['auth_user'])){
    $this->url->redirect('home');
    }
    $attendance = array();
    $class_id=$this->input->post('class_id',TRUE);
    $class_details_arr = $this->api_m->get_class_details($class_id);
    $class_attnd_arr = $this->api_m->fetch_attendance($class_id);
    $x = 1;
    $output = "<h3>Attendance Report for ".$class_details_arr[0]['Class_Name']." on
".$class_details_arr[0]['Class_date']."</h3>";
    $output .= "<table class='table table-hovered table-bordered' style='height:600px;overflow-
y:scroll'><tr><td>Sno</td><td>RegNo</td><td>Student Name</td><td>Department</td><td>Date/Time</td></tr>";
      foreach($class_attnd_arr as $attnd){
        $output .=
"<tr><td>".$x."</td><td>".$attnd['RegNo']."</td><td>".$attnd['student_Name']."</td><td>".$attnd['Department']."</td
>
          <td>".$attnd['logged_date']."</td></tr>";
          $x++;
      }
    $output .= "</table>";
    echo $output;
  }
```

Figure 12 Snapshot of the GET Method of the API Class

3. Results and Discussion

Before Image augmentation, one of the students was not recognized as captured in the database.

AK20/PHS/UND/011	60	2	1
AK20/PHS/UND/017	60	2	1
AK20/PHS/UND/015	60	2	1
AK20/PHS/UND/018	60	2	1
AK20/PHS/UND/013	60	2	1
AK20/PHS/UND/020	60	2	1

Figure 13 Attendance Captured Before Augmentation

After image augmentation, the correct faces were recognized and attendance was marked accordingly.

AK20/PHS/UND/011	60	2	1
AK20/PHS/UND/017	60	2	1
AK20/PHS/UND/015	60	2	1
AK20/PHS/UND/018	60	2	1
AK20/PHS/UND/013	60	2	1
AK20/PHS/UND/020	60	2	1
AK20/PHS/UND/016	60	2	1

Figure 14 Attendance Captured After Augmentation


```
[{"id":"1","RegNo":"AK19\/PHS\/MSC\/007","Class_id":"75","Ses":"2","Sem":"1","logged_date":"2022-07-31 05:06:03",
"student_Name":"Esther Polycarp Sylvester","Department":"Computer Science","State":"AKI","LGA":"IBE","Level":"8",
"Gender":"F","Class_Name":"External Defence","Class_date":"2022-07-29","Lecturer_Name":"Dr Ansa"},{"id":"2",
"RegNo":"AK20\/PHS\/MSC\/002","Class_id":"75","Ses":"2","Sem":"1","logged_date":"2022-07-31 05:06:03",
"student_Name":"Unyime Edet","Department":"Computer Science","State":"AKI","LGA":"UYO","Level":"7","Gender":"F",
"Class_Name":"External Defence","Class_date":"2022-07-29","Lecturer_Name":"Dr Ansa"},{"id":"3","RegNo":"AK19\/PHS\/MSC\/
006","Class_id":"75","Ses":"2","Sem":"1","logged_date":"2022-07-31 05:06:03","student_Name":"Victoria Emmanuel Essien",
"Department":"Computer Science","State":"AKI","LGA":"IBI","Level":"8","Gender":"F","Class_Name":"External Defence",
"Class_date":"2022-07-29","Lecturer_Name":"Dr Ansa"},{"id":"4","RegNo":"AK19\/PHS\/CSC\/MSC\/002","Class_id":"75",
"Ses":"2","Sem":"1","logged_date":"2022-07-31 05:06:03","student_Name":"Ubong David Essien","Department":"Computer
Science","State":"AKI","LGA":"UYO","Level":"8","Gender":"M","Class_Name":"External Defence","Class_date":"2022-07-29",
"Lecturer_Name":"Dr Ansa"},{"id":"5","RegNo":"AK19\/PHS\/MSC\/005","Class_id":"75","Ses":"2","Sem":"1",
"logged_date":"2022-07-31 05:06:03","student_Name":"Samuel Bassey Okon","Department":"Computer Science","State":"AKI",
"LGA":"IBI","Level":"8","Gender":"M","Class_Name":"External Defence","Class_date":"2022-07-29","Lecturer_Name":"Dr Ansa"},
{"id":"6","RegNo":"AK19\/PHS\/MSC\/001","Class_id":"75","Ses":"2","Sem":"1","logged_date":"2022-07-31 05:06:03",
"student_Name":"Mfoniso Polycarp Asuquo","Department":"Computer Science","State":"AKI","LGA":"IKO","Level":"8",
"Gender":"F","Class_Name":"External Defence","Class_date":"2022-07-29","Lecturer_Name":"Dr Ansa"},{"id":"7",
"RegNo":"AK19\/PHS\/MSC\/008","Class_id":"75","Ses":"2","Sem":"1","logged_date":"2022-07-31 05:06:04",
"student_Name":"Mpamugo Ezichi Happiness","Department":"Computer Science","State":"ABI","LGA":"UMU","Level":"8",
"Gender":"F","Class_Name":"External Defence","Class_date":"2022-07-29","Lecturer_Name":"Dr Ansa"}
```


Figure 15 JavaScript Object Notation (JSON) Response from the API

Sno	RegNo	Student Name	Department	Date/Time
1	AK19/PHS/MSC/007	Esther Polycarp Sylvester	Computer Science	2022-07-31 05:06:03
2	AK20/PHS/MSC/002	Unyime Edet	Computer Science	2022-07-31 05:06:03
3	AK19/PHS/MSC/006	Victoria Emmanuel Essien	Computer Science	2022-07-31 05:06:03
4	AK19/PHS/CSC/MSC/002	Ubong David Essien	Computer Science	2022-07-31 05:06:03
5	AK19/PHS/MSC/005	Samuel Bassey Okon	Computer Science	2022-07-31 05:06:03
6	AK19/PHS/MSC/001	Mfoniso Polycarp Asuquo	Computer Science	2022-07-31 05:06:03
7	AK19/PHS/MSC/008	Mpamugo Ezichi Happiness	Computer Science	2022-07-31 05:06:04

Figure 16 Sample Generated Attendance Report

4. Conclusion

We have developed an attendance system that is secure. This system can be easily implemented and it is scalable to accommodate varying sizes of audience. Access to the attendance records/report is secured and not prone to mutilation by unauthorized users. The web-based API allows access to external authenticated users who may need to make policies from the data stored within the database of the system.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] Arsenovic, M., Sladojevic, S., Anderla, A., & Stefanovic, D. (2017, September). FaceTime—Deep learning-based face recognition attendance system. In 2017 IEEE 15th International symposium on intelligent systems and informatics (SISY) (pp. 000053-000058).

[2] Ayodele, O. D. (2017). Class attendance and academic performance of second year university students in an organic chemistry course. African Journal of Chemical Education, 7(1), 63-75.

[3] Bhatti, K. L., Mughal, L., Khuhawar, F. Y. & Memon, S. A. (2018). Smart attendance management system using face recognition. EAI Endorsed Transactions on Creative Technologies, 5(17).

[4] Credé, M., Roch, S. G., & Kieszczynka, U. M. (2010). Class attendance in college: A meta-analytic review of the relationship of class attendance with grades and student characteristics. Review of Educational Research, 80(2), 272-295.

[5] D'Silva, K., Shanbhag, S., Chaudhari, A. & Patil, M. P. (2019). Spot me-a smart attendance system based on face recognition. Int. Res. J. Eng. Technol. (IRJET), 6(3), 4239.

[6] Essien, U., & Ansa, G. (2023). Attendance Based Systems for Face Recognition using a Camera Rail Approach. Researchers Journal of Science and Technology, 3(1), 14–28. Retrieved from https://www.rejost.com.ng/index.php/home/article/view/45

[7] Gomes, C., Chanchal, S., Desai, T. & Jadhav, D. (2020). Class Attendance Management System using Facial Recognition. In ITM Web of Conferences (Vol. 32, p. 02001). EDP Sciences.

[8] Ghosh, D. (2021). Real- Time Attendance System Using Face Recognition Technique. International Journal Of Engineering Applied Sciences And Technology, 5(9). doi: 10.33564/ijeast. 2021.v05i09.043

[9] Jogin, M., Madhulika, M. S., Divya, G. D., Meghana, R. K., & Apoorva, S. (2018, May). Feature extraction using convolution neural networks (CNN) and deep learning. In 2018 3rd IEEE international conference on recent trends in electronics, information & communication technology (RTEICT) (pp. 2319-2323).

[10] Ku, H., & Dong, W. (2020). Face Recognition Based on MTCNN and Convolutional Neural Network. Frontiers in Signal Processing, 4(1), 37-42.

[11] Manju, D. & Radha, V. (2020). A novel approach for pose invariant face recognition in surveillance videos. Procedia Computer Science, 167, 890-899.

[12] Peng, P., Portugal, I., Alencar, P., & Cowan, D. (2021). A face recognition software framework based on principal component analysis. Plos one, 16(7), e0254965.

[13] Raj, A., Raj, A., & Ahmad, I. (2021). Smart Attendance Monitoring System with Computer Vision Using IOT. Journal of Mobile Multimedia, 115-126.

[14] Satpute, R., Sontakke, S., Gondaliya, B., Sonawane, T. & Suryawanshi, K. (2020). Attendance Management System Using Face Recognition. biometrics, 7(05).

[15] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815-823).

[16] Serengil, S. I., & Ozpinar, A. (2021). HyperExtended LightFace: A Facial Attribute Analysis Framework. 2021 International Conference on Engineering and Emerging Technologies (ICEET), 1–4. doi:10.1109/ICEET53442.2021.9659697

[17] Sufyanu, Z., Mohamad, F. S., Yusuf, A. A., Musa, A. N. & Abdulkadir, R. U. (2016). Feature extraction methods for face recognition. International journal of applied engineering research (IRAER), 5, 5658-5668.

[18] Taigman, Y., Yang, M., Ranzato, M. A., & Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1701-1708).

[19] Trigueros, D. S., Meng, L., & Hartnett, M. (2018). Face recognition: From traditional to deep learning methods. arXiv preprint arXiv:1811.00116.

[20] Tuncer, T., Aydemir, E., Ozyurt, F., & Dogan, S. (2021). A deep feature warehouse and iterative MRMR based handwritten signature verification method. Multimedia Tools and Applications, 1-15.

[21] William, I., Rachmawanto, E. H., Santoso, H. A., & Sari, C. A. (2019, October). Face recognition using facenet (survey, performance test, and comparison). In 2019 fourth international conference on informatics and computing (ICIC) (pp. 1-6).

[22] Xia, P., Zhang, L., & Li, F. (2015). Learning similarity with cosine similarity ensemble. Information Sciences, 307, 39-52.

[23] Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE signal processing letters, 23(10), 1499-1503.