**Face Recognition Based Attendance System**

**LLD**

**Contents**

|  |  |  |
| --- | --- | --- |
| SNo. | Content | Page |
| **1** | **Abstract** | **3** |
| **2** | **Problem Statement Description** | **3** |
| **3** | **Objectives** | **3** |
| **4** | **System Architecture** | **4** |
| **5** | **Testing** | **7** |
| **6** | **Pipeline** | **8** |
| **7** | **Q&A** | **9** |

1. Abstract

Nowadays, face recognition systems are prevalent due to its simplicity and awesome performance. For instance, airport protection systems and FBI use face recognition for criminal investigations by tracking suspects, missing children and drug activities.

It is an important factor in any organization. With the advancement of the Deep learning technology the machine automatically detects image and maintains a record of those collected data.

Based on this, face recognition is one of the most secure and advanced systems in today’s life.

Facial Recognition system is a biometric technology that uses different facial features to identify a person.

Face recognition system is used to automatically identify a person through a digital image.

1. Problem Statement Description

Traditionally attendance of employees are marked using the biometrics machines using techniques like fingerprints verification technique, Retina scanning technique or face detection technique. In all this techniques employee has to physically interact with the machine installed to mark his attendance.

The target of our project is to create an state of art facial based attendance system for a scalable organization which can leverage the use of IP cameras by tracking a person face in real time and mark the attendance. Project can also be used in other application areas like **Home Surveillance, Track attendants at busy places, Find Persons, Unlock Phones, Smart Advertisements.**

1. High Level Objectives
2. Register the person and capture face image.
3. Detect the face and Recognize in real time using IP Camera.
4. Mark the attendance.
5. Manage logging.
6. System Architecture

Facial Features

Facial Features

Face Extraction

Validations

**Data Ingestion**

**Data Ready to Train**

Facial Features

Export data from DB

Split X & Y

**Model Building**

DNN Generation

Fine Tuning

**Deployment**

The overall workflow that can be used to achieve the objective would be divided into 4 parts

1. Data validation
2. Data Preprocessing and Data Ingestion
3. Model building
4. Deployment
   * + 1. **Data Validation:** The role of data validation is to validate the image data given by the user against the conditions and make the data available for Preprocessing step.

Method definitions for Data Validations step

|  |  |  |
| --- | --- | --- |
| **Validations** | **Expected** | **Final** |
| Image dimensions | 2 | After discussion |
| Image channels | 3 (RGB) | After discussion |
| File extension | .jpg , | After discussion |
| No of person /image | 1 | After discussion |

* + - 1. **Data Preprocessing and Data Ingestion:** After we have validated the data provided now we need to preprocess the image data for which
  1. we segment the face from the image using MTCNN,
  2. then crop image into 160\*160 pixel array and
  3. then using facenet we convert our image into 512 dimensional vector and

After Preprocessing we need to encode names and attach with the respective vector-

* 1. Label Encoder
  2. One-Hot Encoding

**Method Definitions for Data Preprocessing**

|  |  |  |
| --- | --- | --- |
| **Steps** | **Expected Model** | **Final** |
| Face Extraction | MTCNN , SSD , Object detection model | After Discussion (MTCNN preferred) |
| Facial Features | FaceNet , Transfer learning models | After Discussion (Facenet preferred) |
| Database | MySQL | After Discussion(SQL Preferred) |

1. **Model Building** : After all the data set for the model training. We have to create a dynamic artificial neural network because we don’t know how many labels we get. Hence ANN should have the ability to create according to the dataset. The things we need to take care about ANN will be :

1. End layer dense number

2. Activation function

3. How many layers

4 . How many neurons in each layer.

5. Optimizer

**Method Definitions for Model Training**

|  |  |  |
| --- | --- | --- |
| **Steps** | **Expected Model** | **Final** |
| Best Layers | Keras Tuner for processing best layers, learning rate, #activation functions. | After Discussion (Keras Tuner) |
| Optimizer | ADAM | After Discussion (ADAM preferred) |
| Best Epoch | Find the best #epoch using the best layers such that model does not overfit |  |
| Final Train | Train the final model using the best parameter learnt from the above steps |  |

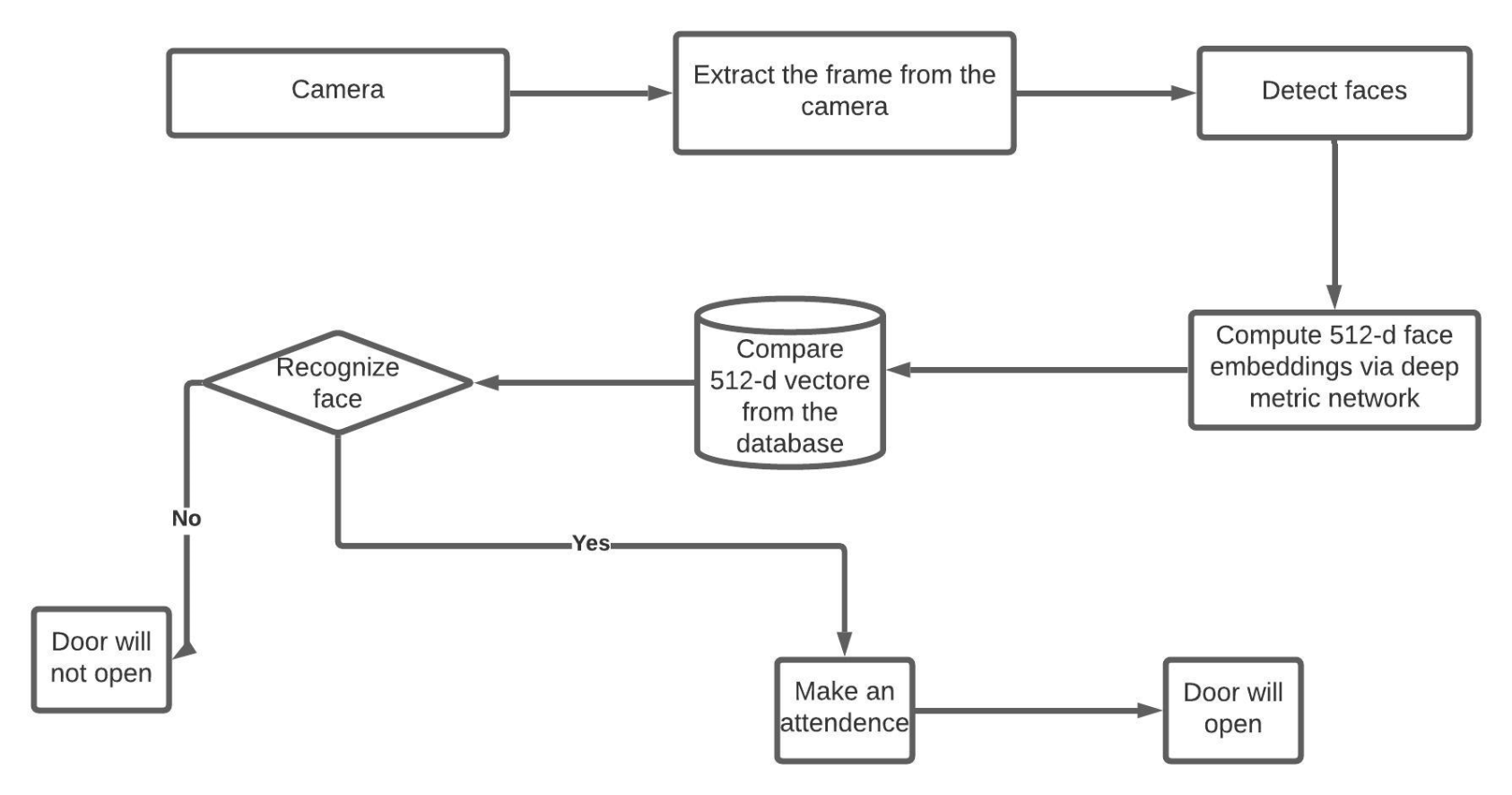
1. **API and Deployment :** So after model building we have to host our model on to the cloud, there can be two ways (i) expose RESTFUL API of the model to the web. (ii) host the model inside the web app

For our model we are using Django to construct web API which have the following features:

* 1. Create Separate Clients for the application
  2. Add Member, Delete Member, Revoke Member
  3. Add Member for training
  4. Real time prediction, Single image Prediction, Batch image Prediction and Manual Prediction
  5. View model logs, Application Logs, View Attendance

|  |  |  |
| --- | --- | --- |
| **Steps** | **Expected Model** | **Final** |
| Web API | Django is currently being used to host the Attendance system | After Discussion between Django vs Flask |

1. Testing
   1. Evaluation Metric – Accuracy
   2. Test Cases
      1. **Different angles of the face** : To be able to detect for large angles.
      2. **Head moving** : To be able to detect for quick movements of head.
      3. **Obstruction of face** : To be able to detect of obstruction such as mask.
      4. **Different lighting conditions** : To be able to detect in very low light and when a light source is right behind the person.
2. PipeLine
3. Extract the frames/Image from the camera
4. Then detect the face from the frame using MTCNN
5. Then convert face into 512-d face embedding via facenet embedding
6. Compare the 512-d vector of the image with the existing image in the database
7. If the face is recognized then the door will be unlocked otherwise the door will not open.

****