Attendance System using Face Recognition

CS337 (Artificial Intelligence & Machine Learning)

Aaryan Gupta, Sarthak Mittal, Shikhar Mundra & Utkarsh Ranjan

Dept. of Computer Science & Engineering, IIT Bombay

November 24, 2022

Overview

Overview

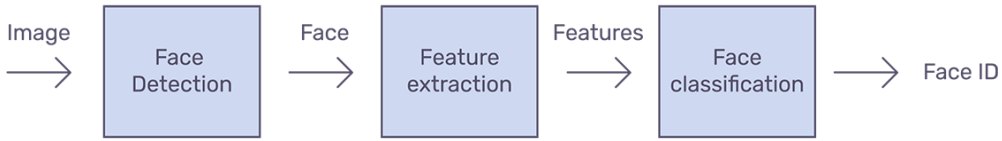
Introduction

"Deep learning advancements in recent years have enabled widespread use of face recognition technology. This project tries to exploit deep learning models used for face recognition and introduces a simple framework for creating and using a custom face recognition attendance system."

This project is based on building an automated attendance system using face recognition. We make use of multiple models chained together to achieve this. Our framework consists of a pre-trained deep neural network Caffe model (ref. [6]) for face detection within an image, deployed using OpenCV DNN configuration (ref. [5]), followed by alignment using Haar Cascade Eye Detector (ref. [8]). This is followed by use of Inception-ResNet v1 architecture (ref. [4]) combined with Triplet Loss (ref. [9]). We conclude by checking classification using Softmax.

Overview

Face recognition can be divided into multiple steps, which can be seen in the pipeline below:



Face recognition pipeline

We perform face detection using DNN (Caffe Model), alignment using Cascade Classifier, feature extraction using Inception-ResNet and classification using Softmax Classifier. We thank Luka Dulcic at Ars Futura (ref. [1]) for his article.

Sample Input

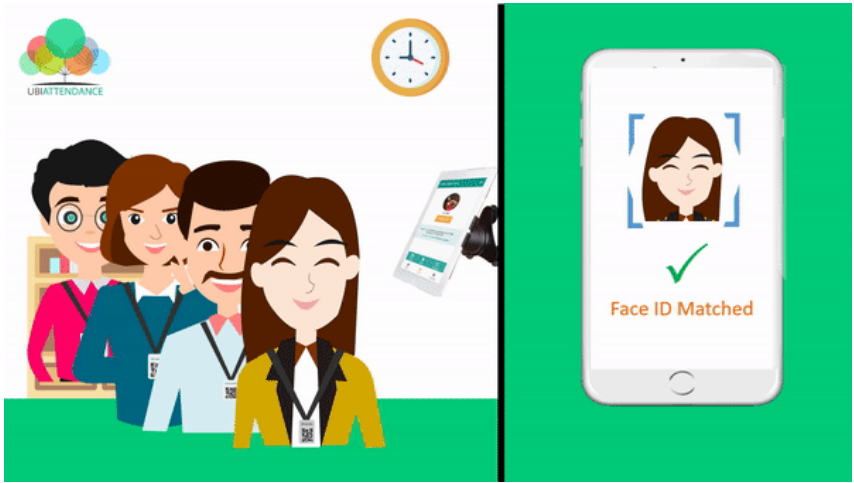
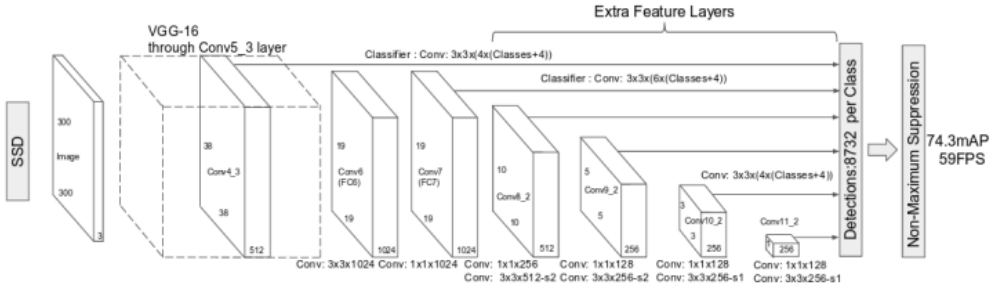


Figure 1: A sample marking of automated attendance

Overview

Detection using DNN Caffe Model

Face detection is an early stage of a face recognition pipeline. It plays a pivotal role in pipelines. Herein, deep learning based approach handles it more accurate and faster than traditional methods. We make use of an external pre-trained ResNet model (ref. [3]) by OpenCV. The model structure is as shown below:



ResNet SSD

Alignment using Haar Cascade Classifier

The method consists of primarily three steps:

- new representation (integral image) that allows for very fast feature evaluation
- classifying by selecting a small number of important features (using AdaBoost)
- cascading successively more complex classifiers which dramatically increases the speed of the detector by focusing attention on promising regions of the image

The object detection model uses some simple features to classify instead of pixels. We use three kind of features:

1. two-rectangle feature, the difference between the sum of the pixels within two rectangular regions
2. three-rectangle feature, difference of sum in center rectangle and sum within two outside rectangles
3. four-rectangle feature, the difference between diagonal pairs of rectangles

Feature Extraction using Inception-ResNet

The Inception-ResNet network is a hybrid network inspired both by inception and the performance of resnet. The key functionality of the Inception-ResNet is that the output of the inception module is added to the input (i.e.data from previous layer).

Just like how FaceNet (ref. [7]) takes an image of the person's face as input and outputs a vector of 128 numbers which represent the most important features of a face (called an embedding), Inception-ResNet also creates embeddings. All the important information from an image is embedded into this vector. Ideally, embeddings of similar faces are also similar.

$$\sum_{i=1}^{N} \left[(f_i^a - f_i^p)^2 - (f_i^a - f_i^n)^2 + N \right]$$

Triplet loss formula

MTCNN

Multi-task Cascaded Convolutional Networks are used to to detect faces in an image and returns the rectangle around a face. It consists of three stages:

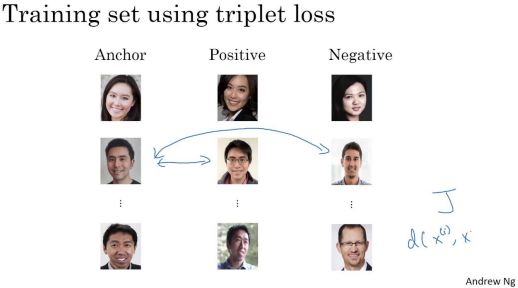
1. P-Net (Proposal Network)
2. R-Net (Refine Network)
3. O-Net (Output Network)

MTCNN tries to achieve three tasks:

1. Face Classification
2. Bounding Box Regression
3. Facial Landmark localisation

Learning Embedding Classification

We pose the problem as a similarity learning problem instead of a classification problem. Here the network is trained to output a distance which is small if the image belongs to a known person and large if the image belongs to an unknown person. However, if we want to output the closest images to a given image, we would like to learn a ranking and not just a similarity.



Training with triplet loss

Overview

Conclusion

Softmax classifier is used as a final step to classify a person based on a face embedding. Softmax was a logical choice since the entire stack is neural networks based, but if the face embeddings themselves are good, other classifiers such as KNN, SVM, Random Forest, etc. should also perform well at this step (ref. [1]).

The combined model functions like an attendance system, and is able to classify images of each person in the validation data into cluster of images of the same person as per the training data. The data that we used, a subset of the LFW Face Dataset (ref. [2]), can be found here in the project repository. We pre-processed the input images (using Face Detection and Eye Alignment) and then trained the Softmax Classification model on the training data along with the Inception-ResNet model (using Triplet Loss). We achieved around **98**% accuracy in training, and $\mathbf{76} - \mathbf{79}\%$ accuracy in validation.

References I

[1] L. Dulcic. Face recognition with facenet and mtcnn, 2019. URL `https://arsfutura.com/magazine/face-recognition-with-facenet-and-mtcnn/`.

[2] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[3] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[4] N. K. Kankam. Understanding inception-resnet v1. URL `https://iq.opengenus.org/inception-resnet-v1/`.

[5] OpenCV. OpenCV DNN Face Detector deployment definition, 2019. URL `https://github.com/opencv/opencv/tree/3.4/samples/dnn/face_detector`.

References II

[6] A. Rybnikov. OpenCV DNN Face Detector caffe model, 2017. URL https://github.com/opencv/opencv_3rdparty/tree/dnn_samples_face_detector_20170830.

[7] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015. URL http://arxiv.org/abs/1503.03832.

[8] O. D. Suarez. OpenCV Haar Cascade eye detector, 2010. URL https://github.com/kipr/opencv/tree/master/data/haarcascades.

[9] Wikipedia contributors. Triplet loss — Wikipedia, the free encyclopedia, 2022. URL https://en.wikipedia.org/w/index.php?title=Triplet_loss&oldid=1096016823. [Online; accessed 24-November-2022].