Real Time Face Recognition Using Face Net

Submitted by:

Burra Mastan Vedasai Sahitya Nithin **&** Sreya Sree Dokala

Department of Electrical Engineering

Central University of Karnataka, Kalaburagi,

Karnataka, India – 585367

Submitted to:

Dr. Sri Phani Krishna Karri

Assistant Professor

Department of Electrical and Electronics Engineering

National Institute of Technology Andhra Pradesh, Tadepalligudem,

Andhra Pradesh, India - 534101

***This report is submitted as part of online summer internship at NIT Andhra Pradesh*

Table of Contents

***Abstract*— Face recognition is one of the most interesting and exciting applications of deep learning. Despite of the significant advancement in face recognition in recent years, implementing at a large scale in real time has its own constraints and consequences. This study shows an approach to implement multiple face recognition with good efficiency using recent technique proposed by researchers at google in 2015 known as FaceNet: A Unified embedding for face recognition and Clustering. The embedding from the FaceNet can be used for multiple tasks such as face recognition, verification and clustering. Our idea is to implement this model along with MTCNN for real time face recognition. Face recognition involves face detection, alignment and classification. In the proposed approach the tasks of face detection and alignment are done using MTCNN, which locates the faces in the given image which are further used for generating embeddings using FaceNet model. The FaceNet model embeds the facial image into 128-dimensions and generates 128 embeddings from the same. These embeddings are further used for classification of facial images.**

I. INTRODUCTION

Face recognition involves face detection, alignment and classification which are very essential in real time implementation. Face recognition is a process of classifying a face into a group of trained faces or identifying a face based on available data. Each and every step-in face recognition has its own significance so, it is substantial to ensure that every step has attained its importance. Face recognition has spread its own importance from preventing retail crime to unlocking the phone, tracking attendance and many more. Despite of the significant advancement in face recognition in recent years, implementing efficiently in a large scale provides with many challenges.

One of the major challenges is detecting multiple faces efficiently in single frame. The main objective of this system is to achieve multiple face recognition in one frame with good efficiency. The frame work uses Google's FaceNet and MTCNN to achieve the goals. Face recognition is achieved through detection, alignment and classification. The task of face detection is done using MTCNN which is known as Multi – Task Cascaded Convolutional Neural Networks. Face detection and alignment are essential to many face applications, such as face recognition and facial expression analysis. However, the large visual variations of faces, such as occlusions, large pose variations and extreme lightings, impose great challenges for these tasks in real world applications.

Prior to MTCNN, the standard technique used for face detection and alignment is cascade face detector proposed by Viola and Jones which utilizes Haar-Like features and AdaBoost to train the cascaded classifiers, which achieve good performance with real-time efficiency [1]. However, quite a few studies indicate that this detector may degrade significantly in real – world applications with larger variations of human faces even with more advanced features and classifiers. The MTCNN frame work overcomes some of these challenges even in real time implementation which makes it more efficient when compared to cascade detector proposed by Viola and Jones.

After detecting and aligning the faces, the features are extracted using FaceNet model. FaceNet, A Unified Embedding for Face Recognition and Clustering is a model which uses a deep CNN and is trained to directly optimize the embedding itself, rather than an intermediate bottleneck layer as in previous deep learning approaches. It is a one-shot model directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. Once this space has been produced, tasks such as face recognition, verification and clustering can be easily implemented using standard techniques with FaceNet embeddings as feature vectors. The FaceNet model is deep CNN of trained on 22 layers with triplet loss as loss function. With these embeddings, faces can be classified based on trained images [2-4].

In early times the traditional face recognition systems were not stable and had several errors in real time implementations. In recent years, advancement in the computation deep learning is being used in face recognition which achieves the state of art.

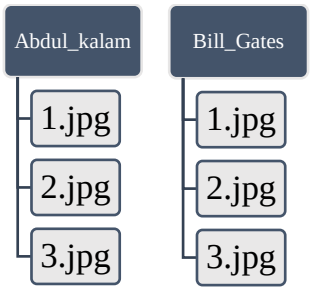
II. APPROACH TO PROBLEM

In spite of many models, only FaceNet is used as it is developed by researchers at Google recently in 2015 which achieved the state of art results. Our approach is to first recognize faces in the images which also uses FaceNet and MTCNN but for classification it uses Support Vector Machines [5]. After this, real time implementation of face recognition has been done but unlike image classification, in real time classification the frames are classified by comparing the embeddings of FaceNet with those of embeddings saved while training without using any technique like SVM. The idea is to deploy this system in a real time application like an attendance system. So, at last the add – on to

this project is to generate an excel sheet in ".csv" file format with the recognized face names with date and time [6].

III. METHODOLOGY

Before getting started with building the model the data set should be prepared. The root folder should be named as the person's name which contains his/her photographs. Training can be done with 5-8 images per person. The number of root folders depends on the number of different faces that should be trained.



The above picture represents the alignment of a sample data set. After processing the data set into required structure each photo will be driven through MTCNN face detector. The MTCNN frame work adopts a cascaded structure with three stages of carefully designed deep convolutional neural network that predicts face and landmark location in a coarse-to-fine manner.

```
[{'box': [22, 20, 87, 109], 'confidence':
0.9999991655349731, 'keypoints': {'left_eye': (50, 58),
'right_eye': (90, 56), 'nose': (79, 74), 'mouth_left':
(58, 101), 'mouth_right': (93, 99)}}, {'box': [155, 23,
73, 100], 'confidence': 0.9999881982803345, 'keypoints':
{'left_eye': (168, 69), 'right_eye': (197, 62), 'nose':
(180, 87), 'mouth_left': (175, 106), 'mouth_right': (201,
101)}}]
```

Figure 1. Values generated by MTCNN for single face

From these values, eyes are taken as key points and are aligned and normalized. The values after normalization are passed into the pretrained FaceNet model saved in h5 file [7]. This model uses 22-layer deep CNN with triplet loss as the loss function. Triplet Loss decreases the distance between an anchor and a positive input when both of which have the same identity, and increases the distance between the anchor and a negative input for the different identities. The weight of FaceNet is optimized using the triplet loss function, so that it learns to embed facial images into a 128-dimensions and generates 128 embeddings. In order to achieve good results from triplet loss function correct triplet selection is crucial. In the model the triplets are generated from the mini batches while training.

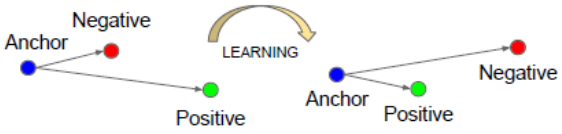


Figure 2. Triplet Loss Function

These 128 embeddings are then saved into a dictionary with each key corresponding to person name and values are 128 embeddings from FaceNet model. This dictionary is saved into a pickle file for further usage. Now, for real time face recognition OpenCV is used to access the web cam. Each frame from live video is again driven through MTCNN face detector and these values are normalized and aligned. The normalized values are used to get encodings from FaceNet model. These values are now compared with those saved in a pickle file while training. Based on the similarity, the face in the frame is classified as particular face [8]. The below picture represents dictionary which was generated while training on the dataset.

Key	Type	Size	Value
Abdul_Kalam	Array of float32	(128,)	[0.05463447 0.01267713 -0.08851028 ... 0.0 ...
Bill_Gates	Array of float32	(128,)	[-0.09949042 0.03731392 -0.06268545 ... 0.0 ...
Christiano_Ronaldo	Array of float32	(128,)	[0.04086207 0.06266457 0.06474748 ... -0.0 ...
Jeff_Bezos	Array of float32	(128,)	[0.02857734 0.02852575 -0.13286741 ... -0.0 ...
Kalyani	Array of float32	(128,)	[0.13524427 -0.08515012 0.13049889 ... -0.1 ...
Mark_Zuckerberg	Array of float32	(128,)	[-0.10641224 -0.06164963 -0.13402186 ... -0.0 ...
Masthani	Array of float32	(128,)	[0.18795219 -0.05068303 0.08869045 ... -0.1 ...
Nithin	Array of float32	(128,)	[0.00419967 0.04910902 0.04281135 ... -0.0 ...
Sachin_Tendulkar	Array of float32	(128,)	[-0.03577474 -0.01182572 0.12390037 ... 0.0 ...
Sundar_Pichai	Array of float32	(128,)	[-0.02541322 -0.11727472 0.02242949 ... 0.0

Figure 3. Dictionary containing embeddings of trained faces

Apart from this, the model generates an excel sheet which contains the name of the recognized person with their respective date and time of recognition [9].

IV. RESULTS AND DISCUSSION

The model built achieved multiple face recognition with good accuracy in live video and was also able to implement the FaceNet model for image recognition. The embeddings from the trained faces are saved into a pickle file which can be used for other tasks without training the model again. For classification, no standard techniques were used. The model saves an

output video of the recognized faces at the speed of 10 frames per second with 640 * 480 frame size.

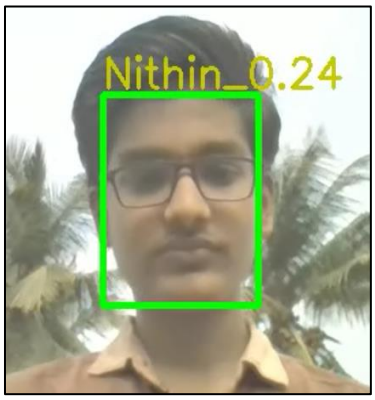


Figure 4. Output of the model

While recognizing, it generates the similarity distance between the trained face and recognized face. This similarity distance has a threshold of 0.4, up to which the recognized face can be considered as the same person and beyond to this it can be safely ignored. The model also generates an excel sheet with the names of the people recognized in the frame, it updates the excel sheet each time we run the model and it appends the names only when a new face appears in the frame. This helps us from the names being repeated. This excel sheet is of ".csv" file format and also contains date and time at which the face is recognized so that it can be useful when the model is deployed in real time application like attendance system.

NAME	DATE	TIME
sreya	08-08-2020	15:05:48
vishal	08-08-2020	15:06:05
vikram	08-08-2020	15:06:16
Nithin	08-08-2020	15:06:27
ashwini	08-08-2020	15:06:40

Figure 5. Sample attendance sheet

This model comes handy to use as we need to just specify some parameters like path for dataset, pretrained face net model and encodings. The user can choose to save the encodings to an external pickle file or not, while executing the model. As mentioned above, there is a need to download the pretrained model from an external source.

Besides, its usage some strict constraint is being laid on the module in the context of dataset and some other related issues. To be more specific about this study, the pros and cons are listed below:

Pros:

- Doesn't require 2-3 python scripts; comes handy in a single script.
- Excel sheet with the recognized names is generated.
- Date and time of faces recognized are also included which can be used for further applications.
- Usage of open source libraries and models makes it free to access and modify.
- Generates 128 embeddings per face in spite of number of images used for training.
- The output frames are stacked and saved as a video which is useful for further reference.

Cons:

- Demands a strict format for data set while training.
- Uses pretrained FaceNet which should be downloaded from an external source.
- There is a need to create an excel sheet of ".csv" file format with specific name before executing the model.
- Restricted to single face in image which are used in dataset for training.
- Uses more time for execution if the data set contains images with great resolution.

For more information and detailed explanation kindly visit our GitHub repository [10].

V. Conclusion

Face recognition is one of the emerging technologies. Besides, it has its own constraints like variation in faces, extreme lightnings which impose great challenges in real time implementation. This module overcomes some of these challenges with MTCNN and FaceNet. These implementation uses latest techniques giving us a chance to overcome with the old techniques like HOG, LBPH etc. Generally, face detection is done using Haar – Cascades but it has less efficiency in real time implementation our approach does the same thing using MTCNN which gave us astonishing results than Haar – Cascades. In recent times, face recognition is being used for attendance system which reduces the human effort. The proposed system can be easily deployed in attendance system with desired modifications which makes the work more flexible.

VI. Acknowledgement

We would express our gratitude to our project guide Dr. Sri Phani Krishna Karri, Assistant Professor, EEE Department, NIT AP, Tadepalligudem for his kind co-operation and encouragement in completing this project. We would express our gratitude to Dr. Vuddanti Sandeep, H.O.D, EEE Department, NIT AP, Tadepalligudem for providing us the internship opportunity. Next, we express our deep and sincere gratitude to our parents for their unparalleled love, help and support. We would also like to thank all who helped us in successfully completing this project. At last, we would thank you, the readers for spending your valuable time reading this report.

References

1. Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao (2015) Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks. In: IEEE Signal Processing Letters https://doi.org/10.1109/LSP.2016.2603342

2. Florian Schroff, Dmitry Kalenichenko, James Philbin (2015) FaceNet: A Unified Embedding for Face Recognition and Clustering. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition https://doi.org/10.1109/CVPR.2015.7298682

3. Online document, Introduction to FaceNet: A Unified Embedding for Face Recognition and Clustering, Analytics Vidhya https://medium.com/analytics-vidhya/introduction-to-facenet-a-unified-embedding-for-face-recognition-and-clustering-dbdac8e6f02

4. Online document, A FaceNet – Style Approach to Facial Recognition on the Google Coral Development board, Towards Data Science https://medium.com/analytics-vidhya/introduction-to-facenet-a-unified-embedding-for-face-recognition-and-clustering-dbdac8e6f02

5. Online document, How to Develop a Face Recognition System Using FaceNet in Keras, Machine Learning Mastery https://machinelearningmastery.com/how-to-develop-a-face-recognition-system-using-facenet-in-keras-and-an-svm-classifier/

6. T. Nyein and A. N. Oo, "University Classroom Attendance System Using FaceNet and Support Vector Machine," *2019 International Conference on Advanced Information Technologies (ICAIT)*, Yangon, Myanmar, 2019, pp. 171-176 https://doi.org/10.1109/AITC.2019.8921316

7. Online document, Facenet, David Sandberg GitHub repository https://drive.google.com/open?id=1pwQ3H4aJ8a6yyJHZkTwtjcL4wYWQb7bn

8. Online document, Practical AI – Face, GitHub repository https://github.com/Practical-AI/Face

9. Online document, Face Recognition and Attendance System, MURTAZA'S WORKSHOP https://www.murtazahassan.com/face-recognition-and-attendance-system/

10. Online document, nithinsahitya (2020) Facerecognition_Using_Facenet, GitHub repository https://github.com/nithinsahitya/Facerecognition_Using_Facenet