Automatic Face Detection and Recognition for Attendance Maintenance

Narayana Darapaneni
Director – AIML
Great Learning/Northwestern University
Illinois, USA
darapaneni@gmail.com

Aruna Kumari Evoori
Student – AIML
Great Learning
Chennai, India
arunakumari.evoori@gmail.com

Vijaya Babu Vemuri
Student – AIML
Great Learning
Chennai, India
vemuri.vijaybabu@gmail.com

Thangaselvi Arichandrapandian
Student – AIML
Great Learning
Chennai, India
selvi.dct@gmail.com

Karthikeyan G
Student – AIML
Great Learning
Chennai, India
karthikeyan1193@gmail.com

Anwesh Reddy Paduri
Research Assistant – AIML
Great Learning
Chennai, India
anwesh@greatlearning.in

Dhivakar Babu
Student – AIML
Great Learning
Chennai, India
dhivakarbabu2208@gmail.com

Jayadev Madhavan
Student – AIML
Great Learning
Chennai, India
jayadevs.16@hotmail.com

2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS) | 978-1-7281-8524-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICIIS51140.2020.9342670

***Abstract*— This paper focuses on building a deep learning based efficient attendance capturing system. Contemporary world is heading towards AI where every second creates a new vision with an enormous change. In Artificial Intelligence (AI), face recognition is one of the fastest growing domains. Instead of using traditional methods for marking attendance, we propose to automate it by identifying human faces with their unique face features known as Face Recognition. Face detection is a prerequisite process for face recognition which aims to identify and locate all faces irrespective of their position, scale, orientation, lighting conditions, expression etc. We created a system architectural solution using YOLO, MTCNN, FaceNet embeddings by applying multiple augmentations, picture quality check and de-noise methods to get a better attendance system with less maintenance, low cost hardware (Google Colab - Free Version), better performance and accuracy.**

Keywords - Face detection, Face recognition, YOLO, Attendance System, FaceNet, Image augmentation.

I.INTRODUCTION

Even today most of the education institutions use conventional methods for marking attendance i.e. by manually. It is usually done by a teacher or by an administrator for each class, and at the end of the day all class attendances are consolidated. Recording attendance and its maintenance is one of the crucial tasks in many of the institutions. Some of these institutions moved to biometric systems like Fingerprint, IRIS, Radio Frequency identification (RFID) to ease out their maintenance. But still these methods have their own drawbacks such as the amount of time taken, cumbersome maintenance of attendance register for classes, chances of fake attendance and also in biometric systems there is always a big challenge of handling huge students' count at once.

Using Deep Learning – Face recognition domain, attendance can be automated. Face detection is a prerequisite step for face recognition. Face detection aims to identify and locate all the faces regardless of their position, scale, orientation, lighting conditions, expressions etc. Face recognition adds identity to the detected face by recognizing from the trained image database with the highest match accuracy. Face recognition provides passive identification i.e. a person to be identified does not need to take any action for their identity. Face recognition reduces the proxy attendance more accurately, than any other techniques. By reading data from live-video from a camera installed in the classroom, converting them to necessary frames, performing face detection on these frames and then applying face recognition on detected faces and finally marking attendance of that person. To get better performance, accuracy and to avoid spoofing, this automated technique can be used at multiple intervals during a typical class hour.

II. EXISTING LITERATURE

To maintain attendance using Face recognition there are several approaches published. Using below references as inferences, we built a model to obtain better accuracy.

Abhishek Pratap Singh et al. [1] proposed a face recognition system using Local Binary Patterns Histogram algorithm and Haar Cascade classifier. LBPH was used for identifying a face and also it recognized both front and side faces irrespective of picture quality, with better recognition rate in real time. The system was able to recognize a known and unknown person. When the rate of change in the frame was very high, occlusions occurred and the proposed method was not able to robustly recognize the faces.

Rudy and Marcus [2] developed a face recognition system that consists of four stage processes. The processes were face detection process using Haar Cascade algorithm and skin color detection where images were converted from RGB format to YCrCb format, alignment process that applied face features normalization, feature extraction process, and classification process using LBPH algorithm. The face recognition accuracy was 98.2% at a face distance 40 cm from the camera with lighting condition (24 lux) and for lighting condition (7 lux) accuracy was 94.7%. Testing was performed for camera distance between 40 cm – 90 cm, accuracy decreased for farther distance face recognition.

Dr.B.R. Shunmugapriya et al. [3], the project presented a face recognition system that used Viola-Jones algorithm for face detection, Discriminative Robust Local Binary Pattern (DRLBP) to extract object texture and edge features, and final step of matching face features with the original image using Euclidean Distance metrics.

Ivanna K. Timotius et al. [4], presented face recognition by combining Generalized Discriminant Analysis (GDA) for face feature extraction and Support Vector Machines (SVM) classifier to classify the faces. Experiment showed that the performance of combining these two methods as a face image classifier was better than by only using SVM. The accuracy obtained from the combined method was above 85%.

Ali Ghofrani et al. [5], the problem of facial expression was addressed in two stages: Face detection and Emotion recognition. For face detection, Multi-Task Convolution Neural Network (MTCNN) accurately detected the boundaries of the face, with minimum residual margins. Emotion recognition leveraged a Shuffle Net V2 architecture which tradeoff between the accuracy and the performance of the model.

Wang Yang and Zheng Jiachun [6] introduced an attendance monitoring system by making use of smartphones available with the class teachers. YOLO - You only look once real-time object algorithm was used for face detection and for face recognition Siamese network was used. The designed system was efficient and reliable as the Siamese network rendered better accuracy in face recognition.

Sunil Aryal et al. [7], proposed an attendance system with combination of facial recognition algorithm and machine learning algorithm. Single Shot Multi-Box Detect (SSD) was used for face detection from an image capturing in real time and recognizing the detected face using 128 embeddings of pre-trained FaceNet model, which are optimized based upon triplet loss. From the experiment analysis the accuracy obtained of the proposed system was 97%. The approach solved the problems of face recognition but cannot identify each and every student present in a class.

III. PROPOSED SOLUTION

The proposed solution uses a Deep Learning approach that involves reading data from live-video from a camera installed in the classroom, converting them to necessary frames, perform face detection on these frames, apply face recognition on detected faces and mark the attendance. To get better performance, accuracy and to avoid spoofing, this automated technique can be used at multiple intervals during a typical class hour and can have interval-configurable as per one's need. This proposed technique can be extended to any kind of use case such as KYC verifications, surveillance, companies' biometric entries, emotional quotient detector, etc.

In our system process, face detection is applied prior to face recognition. Face detection locates and identifies all the faces from an image regardless of face position, expression, scale and image quality like brightness, lighting condition etc. On the detected faces, face recognition adds identity to the face by looking up from the trained image database. Image database consists of the individual face images, which are collected manually.

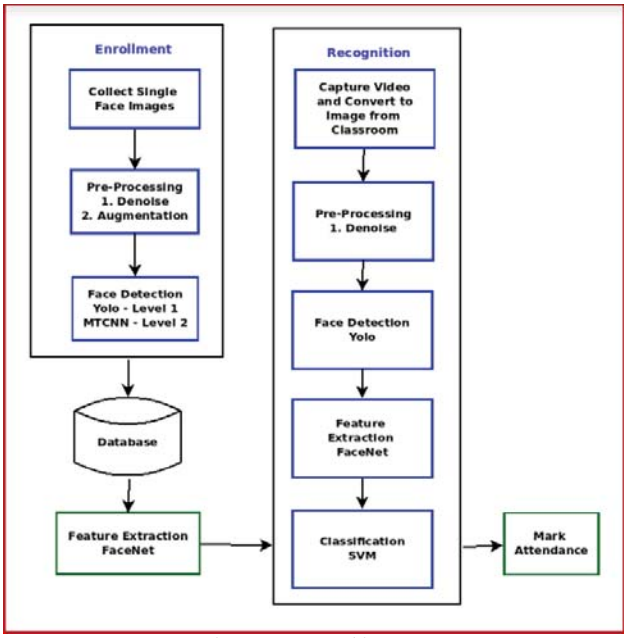

Fig.1. System Architecture

For Face detection we used YOLOv3 & MTCNN models with denoise and quality check methods. And then pre-trained FaceNet model was used for extracting face embeddings from the detected faces. Using these embeddings, we trained our model with linear support vector machine-classifier (SVC) for recognition, as shown in Figure [1] & Figure [2]. We applied a threshold of 50% to assure that the face to be correctly identified, else the face is marked as unrecognized.

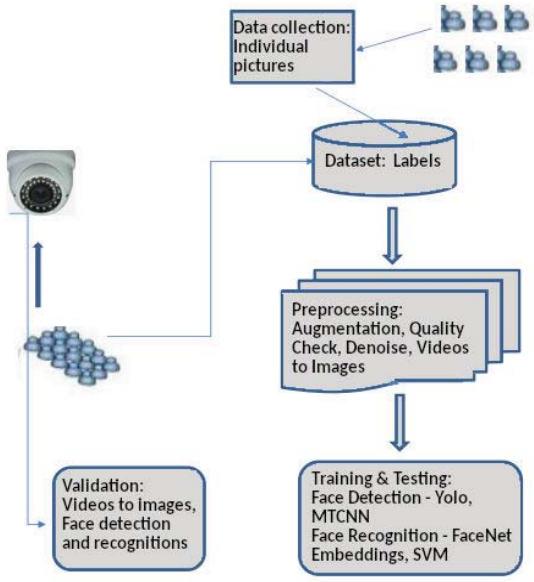

Fig.2. Functional Diagram

IV. DATA GATHERING AND PREPARATION

Dataset was prepared manually by taking images & videos of each individual/classmate separately and converting videos to frames using a python code. These images are then used for model's training, testing and validation after applying image augmentations.

Augmentation:
One of the most arduous tasks for a good face recognition model is to have good quality of adequate data containing faces with various orientation, lighting, exposure etc. The quality and size

of training facial dataset does have a great impact on the results of facial recognition models. Data augmentation comes handy when we need to achieve a robust and enhanced dataset in a limited timeframe. We have used multiple augmentation methods as shown in Figure [3] to improve and have a wide range of image formats in the dataset: · adding random noise · flipping the image · rotating image by -45˚ and +45˚ · adding coolers · adding contrast for brightness · adding morphological operations - erosion and dilation to · landscape and portrait · scaling · occlusion to have semi-face.

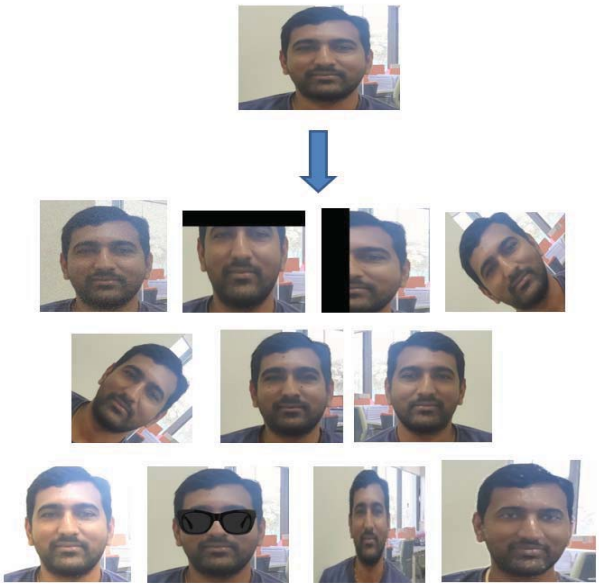

Fig.3.Image Augmentations

V. DATA PREPROCESSING OR WRANGLING

Before extracting necessary face features from an image, to improve its functionality and accuracy, one should use proper pre-processing methods. We applied de-noise filter and quality check assessment preprocessing methods on images.

a) De-noise Filter:

One of the important pre-processing steps in any Face recognition algorithm is an image de-noise. Images with noise may get predicted incorrectly because edges or features of the face will not get extracted properly. The main objective of de-noise method is to completely suppress unwanted noise without degrading the features of the image.

For our model, we applied Non-Local means as de-noise method. This technique will group all similar pixels across the entire image, weigh them and replace the average of those similar pixels in the target cell. This is considered more robust than ordinary local means filter where it will try to replace the pixels with the average of its surrounding pixels. Also, this helps to preserve the edges and features of the image in a more efficient way and acts as a better post-filtering clarity method. Figure [4] shows the de-noise filter input and output.

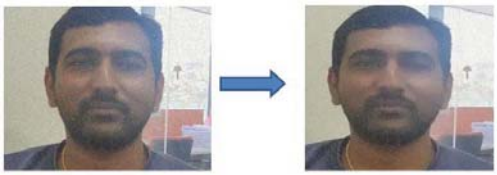

Fig.4. Image de-noise

b) Quality Check:

During our model training, we observed that face detection was also detecting non-facial images and face reflections as actual faces. These detected images are generally in very low quality. We used a preprocessing method called Image Quality Assessment (IQA) to identify the low-quality images and exclude them to improve the accuracy. The performance and accuracy of the recognition algorithms can be examined using these IQA metrics across different fields of computer vision like image compression, image transmission, and image processing. There are two types of Image quality assessment, reference-based and no-reference based evaluation. The main difference is that reference-based methods contain a base image with a high pixel density image which is used to qualify the difference between images. We used "No-reference" based to evaluate the quality of the facial images. This quality check method receives a distorted input image and will not have a reference image to compare it to and assign a value to the input image. The value is then compared to our threshold for the image to be taken or not. Figure [5] quality score used to filter the image

Faces Found: 1

quality check: 15.172774968399466

Faces Found: 3

quality check: 64.45429587176622

quality check: 64.82538665794036

quality check: 87.62925612564786

Fig.5. Image Quality Assessment

VI. EXPLORATORY DATA ANALYSIS

Face detection and recognition in a real time perspective is interesting and challenging. Initially, model training and testing was done on sample images as shown in sample Figure [6,7,8,9] and later on the collected dataset. The dataset consists of videos, frames that were extracted at multiple intervals. These frames are images that consist of faces and objects like laptops, bags, etc. During our model training and testing, noises encountered for face detection are: pose, angle and facial expressions of face, light conditions of the image, size of faces in an image and occlusions.

While detecting faces in an image, handling of above-mentioned noise is pivotal and also detecting faces in a huge group, considering the distances, is a huge challenge. Detected faces from an image are cropped with boundaries having certain width and length. All the images are converted to 416 x 416 pixels in our models to handle the resources and run time. The detected face from an image is then sent to face recognition for labeling. While making the decision to mark attendance, a voting method is used.

We explored other models such as HAAR and LBP, it didn't help us in detecting side faces and the performance was also slow.

We built various models for the proposed solution. For, face detection, MTCNN -multi-task cascaded convolutional network & YOLOv3- You look only once are used and for Face recognition FaceNet embedding with SVM classifier.

i) Face detection:

a) Multi-task Cascaded Convolutional Network (MTCNN):
At times in face detection of straight faces and angled faces in an image, YOLO algorithm was not able to detect faces; and for these undetected faces MTCNN algorithm was used. As MTCNN was performance heavy, we did not use it directly as the main algorithm. In MTCNN architecture, there are three convolution networks called P-Net, R-Net and O-Net. For an input image, P-Net searches for a face, R-Net refines the coordinates received from P-Net and O-Net gives the output of probability of face and co-ordinates of bounding box and facial features (eyes, nose, mouth). MTCNN was able to outperform many of the face detection benchmarks and also still retains the real-time performance. MTCNN was able to detect all the faces of sample images Figure [6] and Figure [7].

Fig.6. MTCNN Detected Faces Sample 1

Fig.7. MTCNN Detected Faces Sample 2(online source)

Observations on MTCNN output:

- MTCNN is able to detect almost all the faces from the dataset provided by us irrespective of face location and face angle.
- It was taking a huge amount of time to run as the architecture is complex.

b) YOLO (You Only Look Once):

YOLO as the abbreviation says it's You Only Look Once is the state-of-the-art and real-time object detection algorithm which uses the convolutional neural network (CNN). In our model, we used YOLOv3, which has feature extractor architecture known as Darknet-53. It has 53 convolution layers and each convolution layer followed by batch normalization and leaky ReLU activation. Weights used in our model are YOLOv3 weights which got trained on the WIDER FACE dataset. Input image size taken is 416 x 416.The input image is divided into grid of dimensions equal to that of the prediction feature map. Example for input image size of 416 x 416 with stride 32, the dimension of cells/feature map will be 13 x 13. The algorithm gives output of a list of bounding boxes with confidence score. YOLO has Non-maximum Suppression (NMS) that ignores the bounding boxes which are less than a certain threshold.

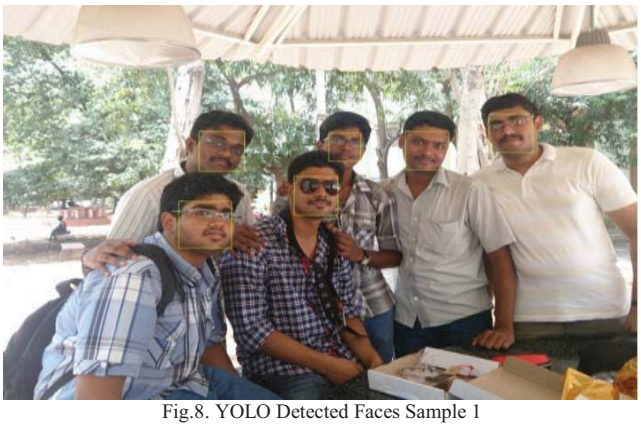

Fig.8. YOLO Detected Faces Sample 1

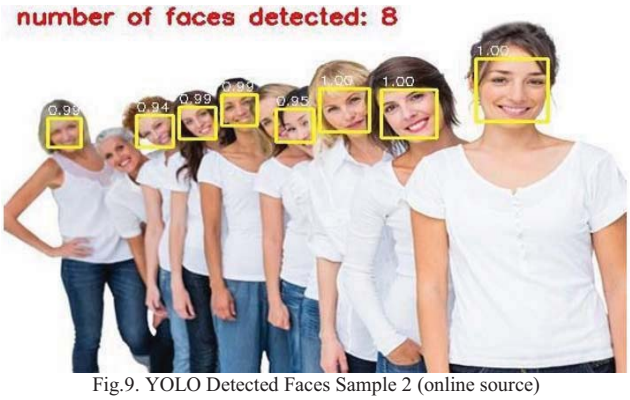

Fig.9. YOLO Detected Faces Sample 2 (online source)

Observations on YOLO output:

- YOLO is able to detect most of the faces from the dataset provided by us. YOLO detected face in samples Figure [8] & Figure [9]
- The best part of YOLO architecture is very fast compared to previous models.
- YOLO fails to detect a few angled faces in an image.

ii) Face recognition:

Face recognition is used to add an identity to the detected face by recognizing faces from the trained image database with highest score/accuracy or with a threshold range, using machine learning algorithms. We used pre-trained FaceNet model embeddings to do face recognition and SVM classifier (SVC) to classify labels.

FaceNet is a unified system for face verification, face recognition and clustering that is built based on learning a Euclidean embedding per image using deep convolutional neural networks. Face verification does face similarity to check if this is the same person, here face similarity is calculated using squared L2 distances, if it is the same person then the distance will be smaller else higher. After the face verification embedding is produced, recognition in FaceNet is done using k-NN classification and clustering using unsupervised learning clustering algorithm. FaceNet has 22-layers that extracts high-quality features from faces and trains its output/features into 128-dimensional embeddings. Triplet loss is the loss function used at the last layer in FaceNet as shown in Figure [10].

Training is done using one image of a face ('anchor'), another image of the same person's face ('positive exemplar'), and an image of a different face ('negative exemplar'). Using triplet loss, the distance between the anchor and the positive is minimized as images are of the same person and the distance between the anchor and a negative is maximized as images are of different person faces. We have used SVM classifier for the face recognition as it can be easily implemented with the FaceNet embeddings as feature vectors.

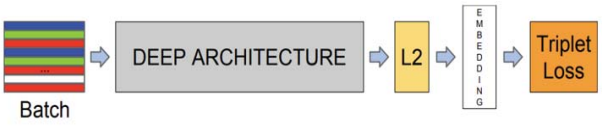

Image Credits: https://medium.com/analytics-vidhya/facenet-architecture-part-1-a062d5d918a1

Fig.10. FaceNet Architecture

Observations on FaceNet and SVM output:

- Euclidean distance is used to find the match or to validate an image from the FaceNet embeddings.
- For image classification, SVM classifier algorithm was used, this is an optional step as FaceNet can classify the image of the person. The reason to use SVC is to identify unrecognized faces with a threshold to avoid the False Negatives.
- While performing hyper parameter tuning in SVC, we observed that for less dataset 'kernel=poly' was providing better results, for large dataset 'kernel=poly' the model was becoming over fit. For our complete dataset, we used 'kernel=linear' as our model provided expected results.

VII. MODEL OUTPUT AND EVALUATION

i) Training and Testing Model:

YOLO and MTCNN with Denoise with FaceNet and SVM:

- TRAINING START TIME: 1581229992.9921224, Local Current: Sun Feb 9 06:33:12 2020
- Total Num of Images Identified 380
- Total Num of Images 415
- Total Num of Class 10
- Train Accuracy: 99.21052631578947
- TRAINING END TIME: 1581243390.9195402, Local Current: Sun Feb 9 10:16:30 2020
- TRAINING EXECUTION TIME: 13397.927417755127
- TEST START TIME: 1581243390.9412613, Local Current: Sun Feb 9 10:16:30 2020
- Test Accuracy: 100.0
- TEST END TIME: 1581244189.782254 Local Current: Sun Feb 9 10:29:49 2020
- TEST EXECUTION TIME: 798.8409926891327 seconds, 13mins

Twins Validation: We ran the model on the twin data (taken the twin celebrity pictures from Google)
Accuracy:
Training Accuracy: 90.625
Test Accuracy: 77.778

ii) Production Model run:
YOLO with Denoise:

- From the Figure [11], Faces detected 13 out of 15 faces in the image.

Fig.11. Input

Fig.12. Output of the detected and recognized faces

- From the Figure [12], Faces recognized correctly: 13
- Faces recognized wrongly: 0
- Faces recognized wrongly as the face is not present in training set: 0

Models Evaluation:
We have evaluated our models using metrics like accuracy, False Negative Rate (FNR), False Positive Rate (FPR).

TABLE.1. RESULTS COMPARISON ACROSS DIFFERENT MODELS

Model	Train Accuracy	Test Accuracy	Time Taken to Train (seconds)	Time Taken to Test (seconds)
MTCNN With FaceNet and SVM	97.163	100.0	17590.681	1147.65
MTCNN with Denoise With FaceNet and SVM	99.581	100.0	36095.058	4215.541
YOLO with FaceNet and SVM	99.014	96.551	4414.308	114.690
YOLO with Denoise with FaceNet and SVM	99.725	100	13820.494	907.389
YOLO and MTCNN with Denoise with FaceNet and SVM	**99.210**	**100**	**13397.927**	**798.84099**

In training, some of the faces were not detected in the YOLO model. For these undetected faces MTCNN model was used. It gave better accuracy but the performance was reduced by a little when compared to YOLO with the de-noise model.

Part No: CFP2058A-ART; ISBN No:978-1-7281-8524-8/20

VIII. IMPLICATIONS

The built face recognition solution can also be used in various use cases like:

- Companies with biometric identification systems can be replaced by face recognition.
- Bank-KYC verification for loan processing or account maintenance. By identifying face using face recognition from profile or proofs with in-live.
- In elections, voters' faces can be recognized as identification proof along with voter ID. Can reduce the chance of electoral fraud.
- Security check-pass in airports can have the face identification to avoid fake IDs.
- Residential or old age homes can use this to alarm for tracking intruders.

IX. LIMITATIONS

- Camera is fixed at a particular location and the classroom covered by the camera lens is alone considered for the attendance. No outdoor sessions were considered.
- The 2D pictures / posters with faces available inside the classroom were not considered.
- Person wearing a mask or a person with cosmetic surgery before and after training are not considered.
- Since the main source of our data is from the camera, power failure without a backup generator to the camera will cause problems for the marking attendance.
- If a new student gets captured in the video who was not part of the training, the model still identifies as someone else and marks attendance for that person.
- The time taken for the models to run are subject to variation as we ran the models in Google Cloud.

X. CLOSING REFLECTIONS

- The designed facial recognition-based attendance system proved to be time saving and efficient with high accuracy.
- Training using YOLO followed by MTCNN (if YOLO is unable to detect face) is very efficient in detecting faces from images.
- The experimental results (Table [1]) shows that the model could recognize student faces and mark their attendances successfully.
- The developed attendance monitoring system using facial recognition proves to be not only secure, reliable, fast but also can significantly improve existing attendance marking activity.
- The system also introduces a combination of using 2 models for face detection where if one model fails to identify a face in the frame, it is passed to the second model which will again try to identify face in the image. These steps increase the total durability of our model and performance in training.
- The developed system uses the concept of the image denoising and quality check which is although time consuming, in turn increases the overall accuracy.
- Ran the entire end to end system process in Google Colab's free version with 12GB RAM, 2vCPU @ 2.2GHz.
- The developed system having YOLO followed by MTCNN model for face detection and FaceNet with SVC for face recognition gave 99% accuracy for 26 labels.

REFERENCES

[1] P. Singh, S. K. S. Manvi, P. Nimbal, and G. K. Shyam, "Face recognition system based on LBPH algorithm," *Ijeat.org*. [Online]. Available: https://www.ijeat.org/wp-content/uploads/papers/v8i5S/E10060585S19.pdf

[2] R. Hartanto and M. N. Adji, "Face recognition for attendance system detection," in *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2018, pp. 376–381.[Online].Available: https://ieeexplore.ieee.org/document/8534942

[3] Dr.B. R. Shunmugapriya and R. S. Sindhu., "An effective attendance management system using face recognition," *Irjet.net*, 2008. [Online]. Available: https://www.irjet.net/archives/V4/i3/IRJET-V4I3142.pdf

[4] I. K. Timotius, The Christiani Linasari, I. Setyawan, and A. A. Febrianto, "Face recognition using support vector machines and generalized discriminant analysis," in *2011 6th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, 2011, pp. 8–10. [Online]. Available: https://ieeexplore.ieee.org/document/6095397

[5] A. Ghofrani, R. M. Toroghi, and S. Ghanbari, "Realtime Face-Detection and Emotion Recognition Using MTCNN and miniShuffleNet V2," in *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*, 2019, pp. 817–821. [Online]. Available: https://ieeexplore.ieee.org/document/8734924

[6] W. Yang and Z. Jiachun, "Real-time face detection based on YOLO," in *2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, 2018, pp. 221–224. [Online]. Available: https://ieeexplore.ieee.org/document/8569109

[7] S. Aryal, R. Singh, A. Sood, and G. Thapa, "Automatic attendance system using deep learning," *SSRN Electron. J.*, 2019. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3352376

[8] "Face Detection Neural Network Structure," *.Mc.ai*, 24-Jul-2018. [Online]. Available: https://mc.ai/face-detection-neural-network-structure/

[9] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks," *arXiv [cs.CV]*, 2016. [Online]. Available: https://arxiv.org/abs/1604.02878

[10] C.-F. Wang, "I implemented a face detection model. Here's how I did it," *Towards Data Science*, 23-Jul-2018. [Online]. Available: https://towardsdatascience.com/mtcnn-face-detection-cdcb20448ce0

[11] C.-F. Wang, "How does A face detection program work? (using neural networks)," *Towards Data Science*, 27-Jul-2018. [Online]. Available: https://towardsdatascience.com/how-does-a-face-detection-program-work-using-neural-networks-17896df8e6f

[12] T. Nguyen, *yoloface*. [Online]. Available: https://github.com/sthanhng/yoloface

[13] A. Kathuria, "What's new in YOLO v3? - Towards Data Science," *Towards Data Science*, 23-Apr-2018. [Online]. Available: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b

[14] R. Balsys, "YOLO v3 theory explained - Analytics Vidhya - Medium," *Analytics Vidhya*, 04-Jul-2019. [Online]. Available: https://medium.com/analytics-vidhya/yolo-v3-theory-explained-33100f6d193

[15] M. Deore, "FaceNet Architecture - Analytics Vidhya - Medium," *Analytics Vidhya*, 04-Apr-2019. [Online]. Available: https://medium.com/analytics-vidhya/facenet-architecture-part-1-a062d5d918a1

[16] D. Kumar, "Introduction to FaceNet: A unified embedding for face recognition and clustering," *Analytics Vidhya*, 26-Jul-2019. [Online]. Available: https://medium.com/analytics-vidhya/introduction-to-facenet-a-unified-embedding-for-face-recognition-and-clustering-dbdac8e6f02

[17] P. F. T. Madio, "A FaceNet-style approach to facial recognition on the Google coral development board," *Towards Data Science*, 28-Aug-2019. [Online]. Available: https://towardsdatascience.com/a-facenet-style-approach-to-facial-recognition-dc0944efe8d1

[18] J. Brownlee, "How to develop a face recognition system using FaceNet in keras," *Machinelearningmastery.com*, 06-Jun-2019. [Online]. Available: https://machinelearningmastery.com/how-to-develop-a-face-recognition-system-using-facenet-in-keras-and-an-svm-classifier/

[19] P. Viola, "Classifier Case Study: Viola-Jones Face Detector," *Psu.edu*. [Online].Available: http://www.cse.psu.edu/~rtc12/CSE586/lectures/violaJonesDetector_6pp.pdf