InnovFaceNet: Deep Face Recognition for Industrial Environments

Nagarjun Gururaj InnovatorsBay Technologies Pvt. Ltd., India
Kanika Batra InnovatorsBay Technologies Pvt. Ltd., India
Email: {nagarjun.gururaj, kanika.batra}@innovatorsbay.com

Abstract

In recent times the usage of intelligent systems have paved way for many applications to be robust and self-reliant. One such popular and vast growing technology is face recognition. Facial Recognition technology is used in security, surveillance, criminal justice systems and many other multimedia platforms. This work proposes a real time facial recognition technology which can be used in any industrial setup eliminating manual supervision, ensuring authorized access to the personnel in the plant. Due to the recent development of COVID-19 pandemic around the world, wearing masks has become a necessity. Our proposed facial recognition technology identifies a person's face with mask or no mask in real time with a speed of 20 FPS on a CPU and an F1-score of 95.07%. This makes our algorithm fast, secure, robust and deployable on a simple personal computer or any edge device at any industrial plant or organization.

1 Introduction

The usage of facial recognition has been very popular and widely used in the computer vision community for identification and several other applications. The first advent of facial recognition algorithms saw a holistic approach to identify human faces. Most popular ones being EigenFaces [1] which used the eigen vectors generated from PCA on the face images to classify them. Other popular holistic approaches include similar techniques, by projecting high dimension data into a lower subspace to identify a face. Later, face recognition systems used a filter based approach in order to identify a face. Most common approach is handcrafting features to identify key attributes of faces followed by some processing and classification techniques [2]. Most popular algorithms include Gabor feature extraction [3] and Local Binary Patterns for face recognition [4]. In early 2010s, learning local based feature descriptors were introduced which indicated a shallow learning of features for face recognition, most popular works of shallow learning include [5], [6]. All the traditional face recognition algorithms suffered from identifying faces with small deviations and fluctuations mainly due to the mismatch of theoretical assumptions with practical scenarios. Traditional face recognition algorithms suffered due to its adaptability to constraints like lighting, pose, non-frontal profile of faces in real time. With advent of deep learning in 2012, the performance of face recognition algorithms have been boosted considerably. The usage of deep networks in effective feature learning across the layers contributed to a significant performance gain. In 2014, DeepFace [7] achieved the SOTA accuracy on the famous LFW benchmark [8], approaching human performance on the unconstrained condition for the first time (DeepFace: 97.35% vs. Human: 97.53%), by training a 9-layer model on 4 million facial images [2]. Recent deep learning networks like FaceNet [9] which use an existing face embedding to identify the faces achieved an accuracy of 99.63% on the LFW benchmark [7] with Siamesenet [10] which uses two deep networks to produce a similarity score to identify the faces, serve as motivation for our proposed algorithm. However, the trade-off between performance and computational size in deep learning algorithms still exists, training these algorithms incur cost and computational resources.

To overcome training with huge resources, the proposed algorithm uses three pre-trained deep networks [face-detection-retail-0005 to detect faces and predict their bounding boxes; landmarks-regression-retail-0009 to predict face keypoints; face-reidentification-retail-0095 to recognize persons] provided by Intel Openvino Platform. The paper is structured as follows: Section 2 provides an overview of the dataset used to test our proposed algorithm; Section 3 discusses the proposed algorithm in brief; Section 4 discusses the working of real time face recognition engine as a single module; Section 5 discusses the results obtained on our dataset and Section 6 discusses the outcomes and possible extensions of the algorithm.

2 Dataset

InnovFaces is a synthetic dataset conceived and prepared by InnovatorsBay Technologies. It is generated by capturing faces from raw videos of 29 personnel recorded on a 480p camera at a manufacturing facility in India. To ensure facial recognition on masked faces, we superimpose a mask on the extracted faces of the personnel. Later we combine these images with the extracted faces without masks. Now, the face database contains extracted face images and the superimposed mask face images. Our original data contains 29 recorded videos. The generated face database consists of 3595 extracted face images with and without mask, each image corresponding to a size of $112 \times 112 \times 3$. Two sample images from InnovFaces is shown in Fig 1. Fig 1a shows the extracted face without mask and Fig 1b shows the extracted face with superimposed mask. We test this algorithm on a real time stream from a camera mounted at the entrance of the facility.

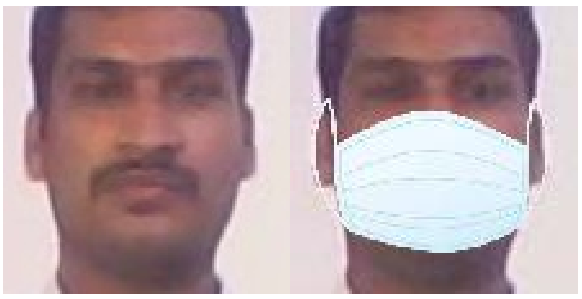

(a) Without mask *(b)* Superimposed mask

Fig. 1: Sample images from InnovFaces

3 Method

In order to design a robust real time algorithm in terms of performance and speed we experimented various algorithms as proof of concept for the proposed algorithm. First, we used a real time face recognition with the Ultralight face detector [11] and MobileFaceNet [12] which produced an above average performance in terms of accuracy with low real time speed trained on two faces. To enhance the accuracy an additional SVM layer with softmax was added. This boosted the accuracy by a very small margin with low speed. To satisfy the speed requirements of a real time processing system, an approximation on the weights was applied using the floating point precision upto 32 bits [FP32] using the Intel Openvino Platform. This helped in accelerating inference on real time feed. We also experimented the dataset containing two faces per participant with the VGGFace [13] algorithm which performed poorly in real time conditions in terms of performance and accuracy.

As a result of applying the above algorithms on dataset containing two faces per participant, factors like lack of training data, speed of the algorithm and performance accuracy was deeply affected. To overcome this, we generated many copies of extracted faces from each class in different pose, frontal modes and lighting conditions. For this purpose, we used the Intel Openvino platform which provided an approximated model for real time inference on a CPU. We used a combined model of three variants, the first model corresponds to a face detector which is shown in Fig 2. The face detector model consists of the conventional MobilenetV2 [14] with an SSD single head attached to it. The single SSD head from 1/16 scale feature map has nine clustered prior. The input to this network is a sequence of frames from the camera which is reshaped to 300 $\times$ 300 $\times$ 3. The output of the MobilenetV2 is fed as an input to SSD head which produces a vector of [1, 1, N, 7] where 'N' is the number of detections. For each detection, the description has the format: [imageid, label, conf, xmin, ymin, xmax, ymax] where imageid is the ID of the image in the batch; label is the predicted class ID; conf is the confidence for the predicted class; (xmin, ymin) are coordinates

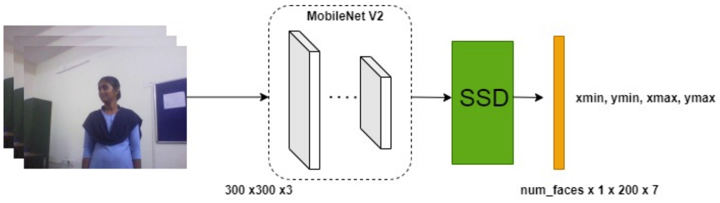


Fig. 2: Face Detector Model

of the top left bounding box corner; (xmax, ymax) are coordinates of the bottom right bounding box corner. Next, we use a custom built model illustrated in Fig. 3 for predicting landmarks of the detected faces in the frames. It has a classic convolutional design: stacked 3x3 convolutions, batch normalizations, PReLU activations, and poolings. Final regression is done by the global depthwise pooling head and FullyConnected layers. The model predicts five facial landmarks: two eyes, nose, and two lip corners. Finally, we use

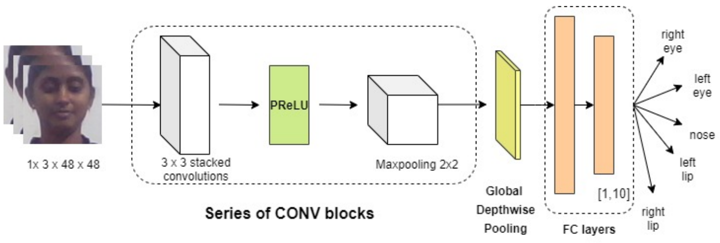


Fig. 3: Landmark Regressor Model

a deep learning model for face re-identification purpose. This is a lightweight network based on MobileNetV2 backbone, which consists of 3x3 inverted residual blocks with squeeze-excitation attention modules. Instead of the ReLU activations used in the original MobileNetV2, this network uses PReLU ones. After the backbone, the network applies global depthwise pooling and then uses 1x1 convolution to create the final embedding vector as shown in Fig.4. The model produces feature vectors which should be close in cosine distance for similar faces and far for different faces. The model

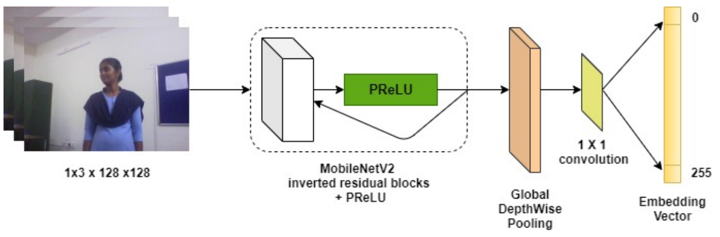


Fig. 4: Face Re-identification Model

performs at LFW benchmark accuracy given that the faces are most aligned in the frontal position. To do this, we use regressed points and the given reference landmarks to build an affine transformation to transform regressed points to the reference ones and apply this transformation to the input face image. The input to this model is an image of size [1 $\times$ 3 $\times$ 128 $\times$ 128] and output is a row embedding vector of shape [1, 256, 1, 1], containing a row-vector of 256 floating point values. Outputs on different images are comparable in cosine distance [15].

4 Putting It All Together

In Section 3, deep networks used to build the face recognition model are discussed. Here, we combine these models to form a single face recognition engine. These models are combined and simulated as a single engine using the Intel Openvino Platform [15]. The face recognition engine illustrated in Fig. 5 reads the specified input video stream frame-by-frame, be it a camera device or a video file, and performs independent analysis of each frame. In order to make predictions, the application deploys the three models on the specified devices, in our case is a CPU using OpenVINO library and runs them in asynchronous manner [16]. An input frame is processed by the face detection model to predict face bounding boxes. Then,

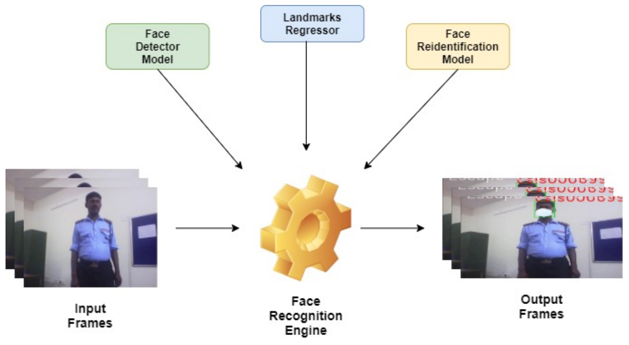


Fig. 5: Comprised Face Recognition Model

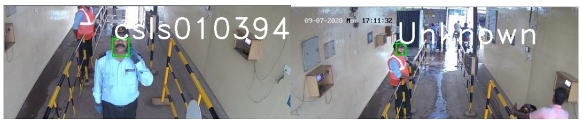
face keypoints are predicted by the corresponding model. The final step in frame processing is done by the face re-identification model, which uses keypoints found to align the faces and the face database to match faces using cosine distance found on a video frame with the ones in the dataset given in equation 1;

$$C_S = \frac{I_{data} \cdot I_{test}}{\|I_{data}\| \times \|I_{test}\|} \tag{1}$$

where 'C_S' represents the cosine similarity distance, 'I_{data}' is the embedding vector generated from the dataset, 'I_{test}' is the embedding vector generated from the test input frames.

5 Results

This section discusses the results obtained by deploying our face algorithm at the manufacturing facility. Before we present the results, there is a huge challenge of assembling a single or multiple personnel at once. In order to test in real-time, we simulated a testing environment at the entrance by assembling all 29 personnel of InnovFaces dataset to walk towards the camera with face masks and without it. An illustration of real time testing results are shown in the Fig. 6 below. We also tested our face recognition on a com-



(a) Authorized face identification *(b)* Unauthorized face identification

Fig. 6: Sample Realtime Results

pletely occluded face and the algorithm fails to detect the face in such scenarios. The primary reason behind this is the missing features which are to be learned by the face detector model for detecting faces in such scenarios. The demonstration of such a scenario is presented in Fig. 7 below. The performance of the algorithm is

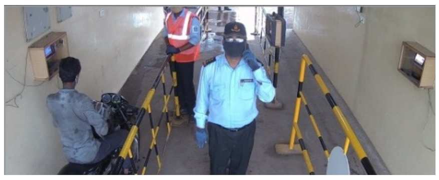


Fig. 7: Real time results for a completely occluded face

measured using F1-Score metric for each personnel's face identity. The general formula for F1-score is given by equation 2 below.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2}$$

Precision corresponds to the total rate of true positives which are identified correctly out of all instances and Recall corresponds to number of true positives which are identified correctly. Precision and Recall are calculated using equation 3 and equation 4.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP+FN} \tag{4}$$

For testing our algorithm, we encode the classes as true class which is the employee ID and false class being "unknown" for unauthorized persons. We also tested the algorithm for its performance with respect to speed. All the inference optimizations have been done using the Intel Openvino Platform by applying a FP32 approximation on the models.

The results are first generated on a dataset containing two faces per participant using the face recognition ultralight model [11], face recognition ultralight model with support vector machine (SVM) and VGGFace model [13]. We also optimized these models for inference speed up in real time. Face Recognition Ultralight Model with SVM utility is ranked second best with F1-Score of 77.59% for two faces. We also test the speed of the algorithms and achieve only 7 FPS at best for these models. We also try to optimize the models in terms of speed by converting them to inference-efficient models using Intel Openvino Toolkit [16] and the speed is boosted by 3 FPS for ultralight model with SVM utility and VGGFace respectively. These models provide a baseline for our algorithm in terms of both speed as well as accuracy, hence we design the proposed algorithm and test it on 29 employees of the organization. The results for the proposed model and the baseline models is depicted in the Table 1 below with our proposed model ranked best, both in terms of speed of 20 FPS and average F1-score of 95.04%.

Algorithm	**Speed (in FPS) [without OpenVino]**	**Speed (in FPS) [with OpenVino]**	**F1-score (in %)**
Face Recognition Ultralight	7	10	76.17
Face Recognition Ultralight + SVM	7	10	77.59
VGGFace	2	5	70.50
InnovFaceNet (Ours)	12	**20**	**95.04**

Table 1: Performance of proposed algorithm and baseline models: InnovFaceNet (best), Face Recognition Ultralight+SVM (second best)

The average F1-score of our proposed algorithm for 29 employees is calculated by taking a simple average of individual F1-scores calculated for each employee in real time. The individual F1-scores for each employee is depicted in Fig. 8. As we see, the performance of our proposed model is more than 95% for most of the employees and 100% for some employees. During the testing, the employees were wearing face masks for some duration and did not wear them for certain duration. We calculate the performance for these combined scenarios. However, the performance of recognizing few employees are low. Employee number 18 has an F1-score of approximately 67% and Employee number 19 has an F1-score of around 50%. The primary reason is the prior insufficient face data for these employees. Due to this, the algorithm does not generalize with the constraints like pose, lighting and frontal profile during test time. The amount of sufficient data plays an important role in generalizing faces in such unconstrained scenarios.

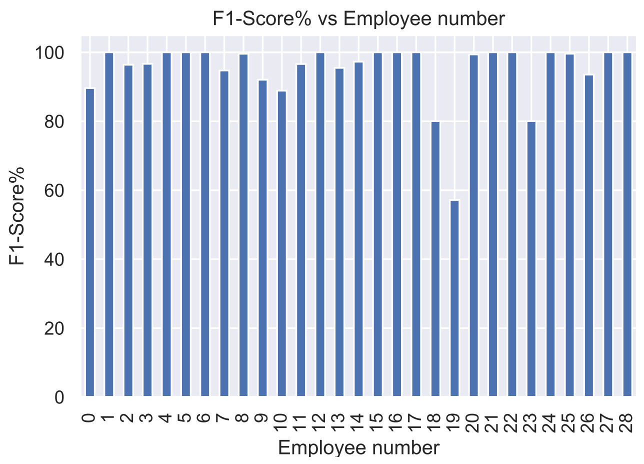


Fig. 8: Individual F1-scores of employees tested real time

6 Conclusion and Future Work

From Section 5, the primary consideration for our proposed algorithm is the importance of sufficient data to match with the detected faces in real time for all unconstrained scenarios. However, our proposed algorithm which uses a combination of three deep neural networks as discussed in Section 3 perform very well with high accuracy in real time. The placement of the camera plays a vital role in enhancing the performance of the algorithm. More the camera angle is mounted or aligned horizontally to face, the results are more accurate as the algorithm is able perform face detection, align face using keypoints and identify the face more efficiently. Another important factor is the distance of the camera from the face, so that it captures the entire face. The reason for this is if there are distant faces from the camera, the face detection algorithm fails to capture the face due to the small area of the face region, hence it is recommended to place the camera at optimum distance. Another key aspect is the quality of the camera which means the camera should be of good resolution, if not the face images are blurry and the network produces erroneous results. Our proposed algorithm can be used at any organization or industry for applications like bio-metric identification, security or surveillance. As the world goes through the pandemic the usage of face masks is inevitable. Our algorithm performs exceedingly well for recognizing faces with face masks. The reason for this is solely due to the amount of synthetic reference data generated by us which captures face images wearing masks in unconstrained conditions. Our algorithm is also deployable at a remote station, organization or a huge industry with just a help of a desktop or a CPU instance which saves time for installation and deployment cost by a huge margin.

Inspite having a very good face recognition algorithm for industrial environments, from a practical point of view one can improve the algorithm and the real time streaming. From an algorithmic view, face mask recognition is not solely dependent on data as discussed before and hence one can extend this algorithm to study the explainable attributes contributing to face mask recognition. In real time, the scaling of InnovFaces dataset plays a key role. As the amount of data increases, the data streaming becomes a bottleneck and introduces a delay in real time. To overcome this, one can experiment with big data engineering platforms like Apache Spark. One can extend this algorithm by generating synthetic data using GANs or other image processing techniques which will boost the accuracy of the algorithm. An effective communication and support from the industry personnel is key to deploy the solution. An interesting prospect would be to deploy these algorithms on edge devices using GPU platforms where one can use the TensorRT library to perform the GPU approximations on the deep learning model to boost the speed of the algorithm.

Acknowledgments

This work would not have been possible without the support of all employees of the manufacturing facility. We would like to particularly thank the 29 employees of the InnovFaces dataset for volunteering and spending time for data recording and their valuable support.

References

[1] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[2] W. Mei and W. Deng, “Deep face recognition: A survey,” *arXiv preprint arXiv:1804.06655*, vol. 1, 2018.

[3] R. Mehrotra, K. R. Namuduri, and N. Ranganathan, “Gabor filter-based edge detection,” *Pattern recognition*, vol. 25, no. 12, pp. 1479–1494, 1992.

[4] T. Ahonen, A. Hadid, and M. Pietikainen, “Face description with local binary patterns: Application to face recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[5] Z. Cao, Q. Yin, X. Tang, and J. Sun, "Face recognition with learning-based descriptor," in *2010 IEEE Computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2707–2714.

[6] Z. Lei and S. Z. Li, "Learning discriminant face descriptor for face recognition," in *Asian Conference on Computer Vision*. Springer, 2012, pp. 748–759.

[7] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," 2015.

[8] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4873–4882.

[9] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[10] Y. Guo and L. Zhang, "One-shot face recognition by promoting underrepresented classes," *arXiv preprint arXiv:1707.05574*, 2017.

[11] linzai, "Face Detector Ultralight," https://github.com/Linzaer/Ultra-Light-Fast-Generic-Face-Detector-1MB, 2019, [Online; accessed 20-July-2020].

[12] S. Chen, Y. Liu, X. Gao, and Z. Han, "Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices," in *Chinese Conference on Biometric Recognition*. Springer, 2018, pp. 428–438.

[13] Z. Qawaqneh, A. A. Mallouh, and B. D. Barkana, "Deep convolutional neural network for age estimation based on vgg-face model," *arXiv preprint arXiv:1709.01664*, 2017.

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[15] Intel, "Face Reidentification Model," https://docs.openvinotoolkit.org/2019_R1/_face_reidentification_retail_0095_description_face_reidentification_retail_0095.html, 2019, [Online; accessed 20-June-2020].

[16] Intel, "Face Recognition Model," https://docs.openvinotoolkit.org/2020.3/_demos_python_demos_face_recognition_demo_README.html#see_also, 2019, [Online; accessed 20-June-2020].