eInfochips
An Arrow Company

ARROW
Five Years Out

Face on Edge: Real Time Face Recognition

Face recognition is a method of identifying or naming individuals/group using facial features. The face recognition system helps to identify people in photos as well as in videos. Recently, these systems can achieve a state of the art performance by utilizing deep learning based approaches. Face recognition systems are widely used in applications like biometric, surveillance, fleet management, retails, social media, etc. In this article, the details of the face recognition system developed and deployed on the edge device (Nvidia Jetson) by eInfochips, are discussed.

Introduction

Face recognition is a method of identifying the name/ID of the faces present in an image or video. This method learns the features of individual faces and differentiates the features from each other. This in turn helps in identifying unique features for each face/class. Facial recognition is one of the top emerging technologies during 2020-25 [1]. Figure 1 shows the predicted growth of facial recognition technology in various sectors like military, government, retail, etc. by TechSci Research group [2].

The facial recognition system (POC) developed by eInfochips focuses on the applications rely on edge devices, which captures face close to the camera (2 to 3 feet distance), with clear face visibility, without occlusion, and with a minimum image resolution of 720 x 640. Deploying the face recognition model in an edge device is challenging than cloud based servers. Since the computational capacity is limited on edge devices, optimizing and fine-tuning the model without degrading its accuracy and performance (Frame rate) is essential. In the present work, the model is optimized using the TensorRT module and deployed on the Nvidia Jetson Xavier AGX board. The developed system is well suited in applications like home security, school/office attendance, driver verification, smart advertising in retails, etc.

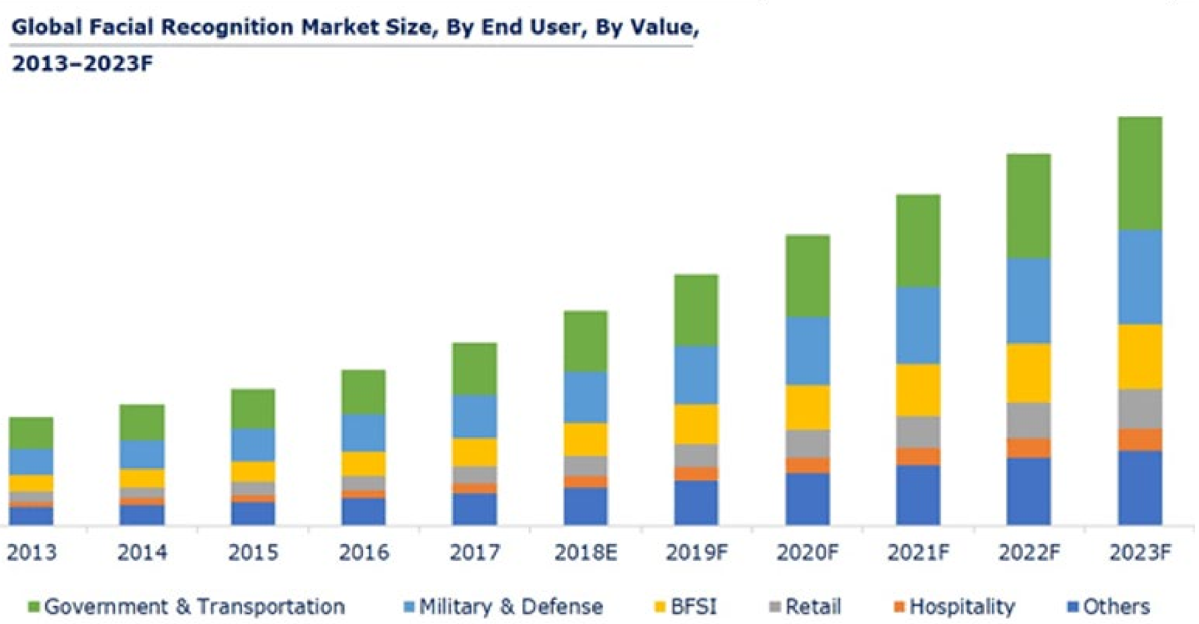


Figure 1: Bar chart shows the growth of face recognition [2]

Facial recognition pipeline

To develop a facial recognition system, three different models are used.

- Face detection – MTCNN (Multi-cascade convolutional neural network) [3]
- Feature extraction – Facenet [4]
- Feature classification – EVM (Extreme value machines) [5]

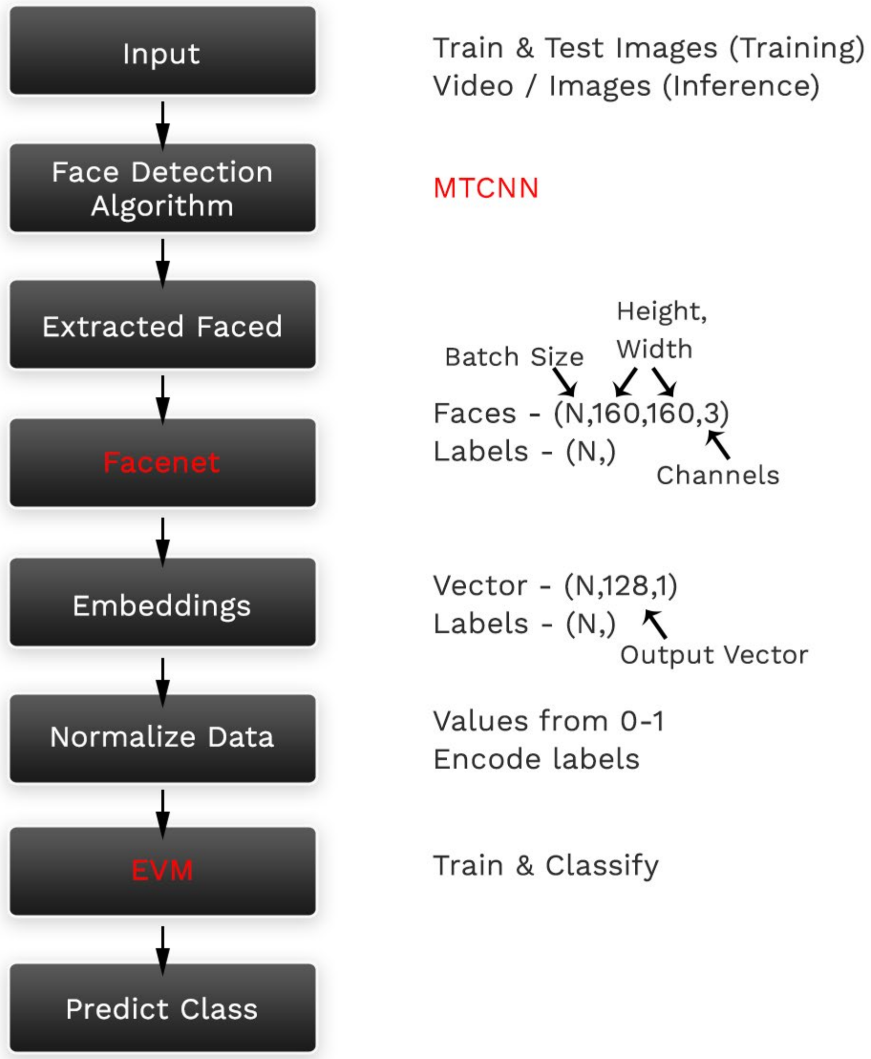


Figure 2: Flow chart of the face recognition pipeline

The face detection/MTCNN model is used to identify faces, which are cropped and further passed to the facenet model. The purpose of the facenet model is to extract unique features from each face. The feature classification algorithm, EVM creates a boundary for each class. Here, the class represents a collection of single/multiple pictures from a single individual. The model is trained on multiple classes and tested for accuracy. Figure 2, shows the flow chart of the face recognition pipeline developed. Let us see the details and working of all the models.

Face detection

Detecting or localizing face in the given image or video is the first stage in the face recognition process. As shown in figure 3, there can be multiple faces and it can be anywhere in the image. The face detection algorithm identifies the face with a confidence value and returns the co-ordinates of those faces. Using the co-ordinates and the confidence value, only the faces are cropped from the image and sent for further processing. A state of art algorithm known as MTCNN is used for this process.

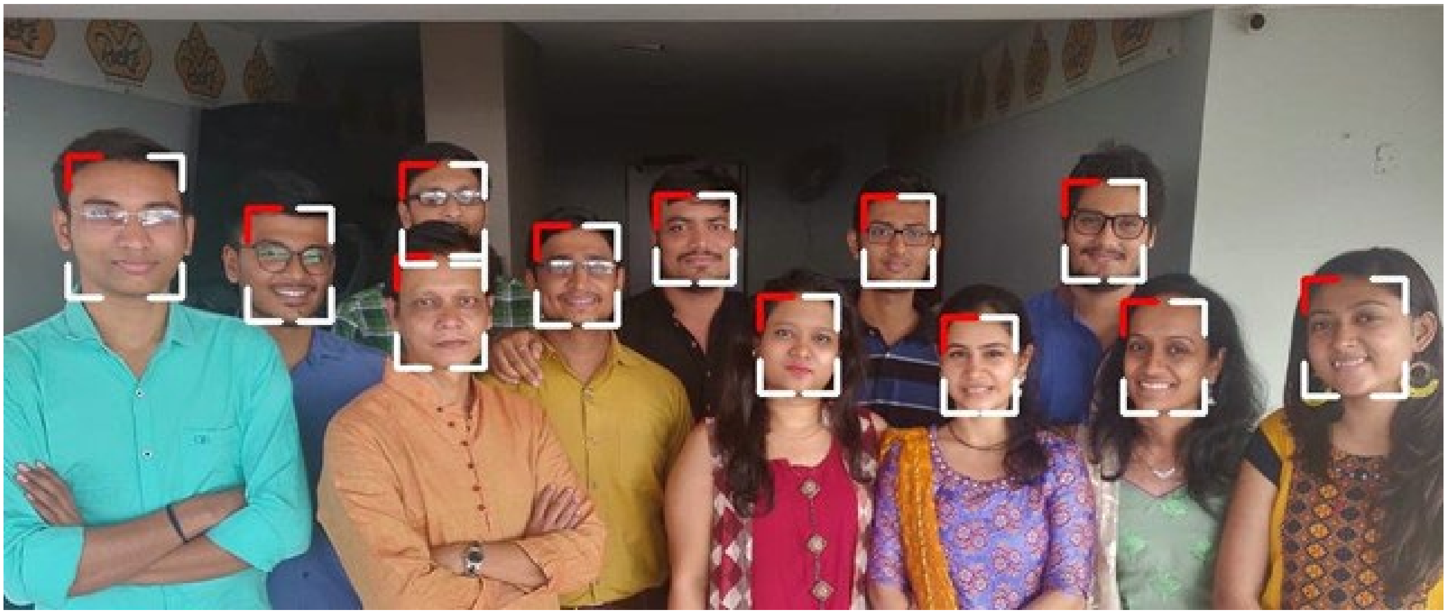


Figure 3: Image showing the MTCNN model identifying the faces

How does MTCNN work?

MTCNN algorithm has three Convolutional Neural Networks (CNN) known as Proposal network (P-net), Refine network (R-net), and Output network (O-net). A sliding window of size 12 x 12 x 3 passes over the input image, extracts features, and passes to convolutional layers of P-net. As shown in figure 4, the P-net layer proposes the location of the faces (bounding box regression layer) with its confidence values (face classification layer). Along with this, the P-net also gives the co-ordinates of eyes, nose, and mouth (facial landmark localization layer). The output of the p-net has false positives, refined by passing it to R-net. The outputs of the R-net are further passed to O-net to get the final confidence values and face co-ordinates.

Feature extraction

Facenet is a CNN based state of art model, developed by Google Inc. Figure 5 shows the sample architecture of the facenet model. After identifying the faces using MTCNN, each face is cropped and passed to the facenet model for feature extraction. Since each face appears different from the other, a unique feature can be extracted for each face. These features are represented as a one-dimensional vector (embeddings) of shape N x 1 x 128, where N is the batch size as shown in figure 2.

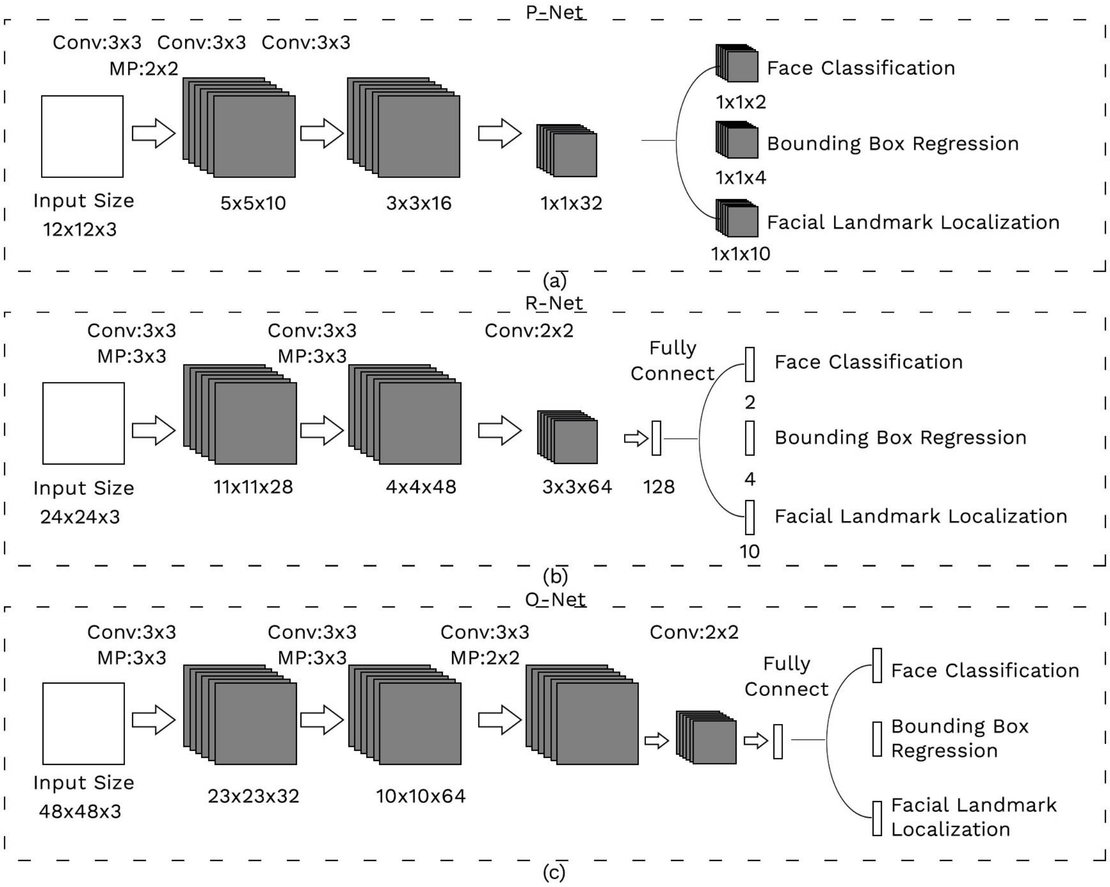


Figure 4: Convolutional neural networks used in MTCNN model [3]

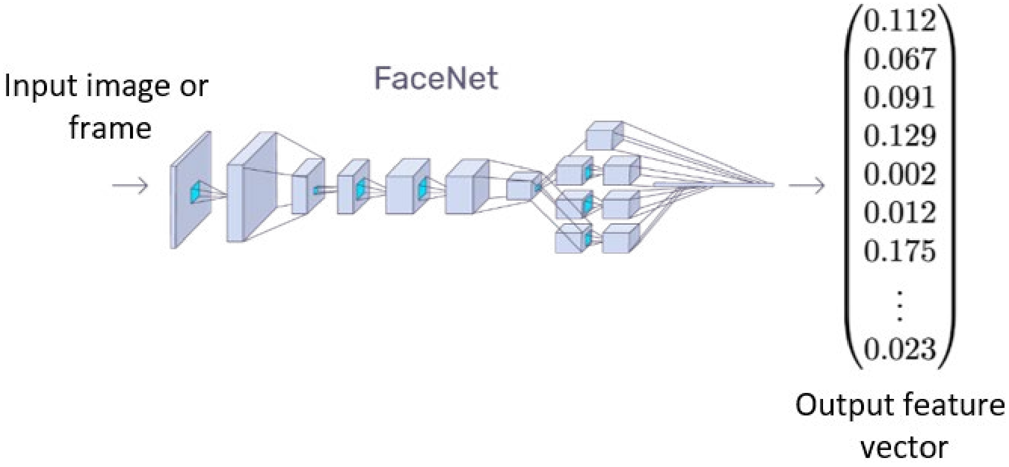


Figure 5: Facenet model architecture showing the inputs and outputs [4]

Feature classification

Features extracted from all the images representing a particular class will be similar. For example, figure 6 shows five different classes represented by red stars, violet squares, green diamonds, A-dots, and question mark "?". It can be observed that the red stars are close to each other as it all belongs to a single class. Similarly, the diamonds, squares are all close to each other. During the training process, the EVM model creates a boundary for each class separating each other. During inference, the feature vector either mapped to any of these circles or outside the circle. If the feature vector falls in any of these circles, it is labeled to that particular class, or else it is labeled as an unknown class.

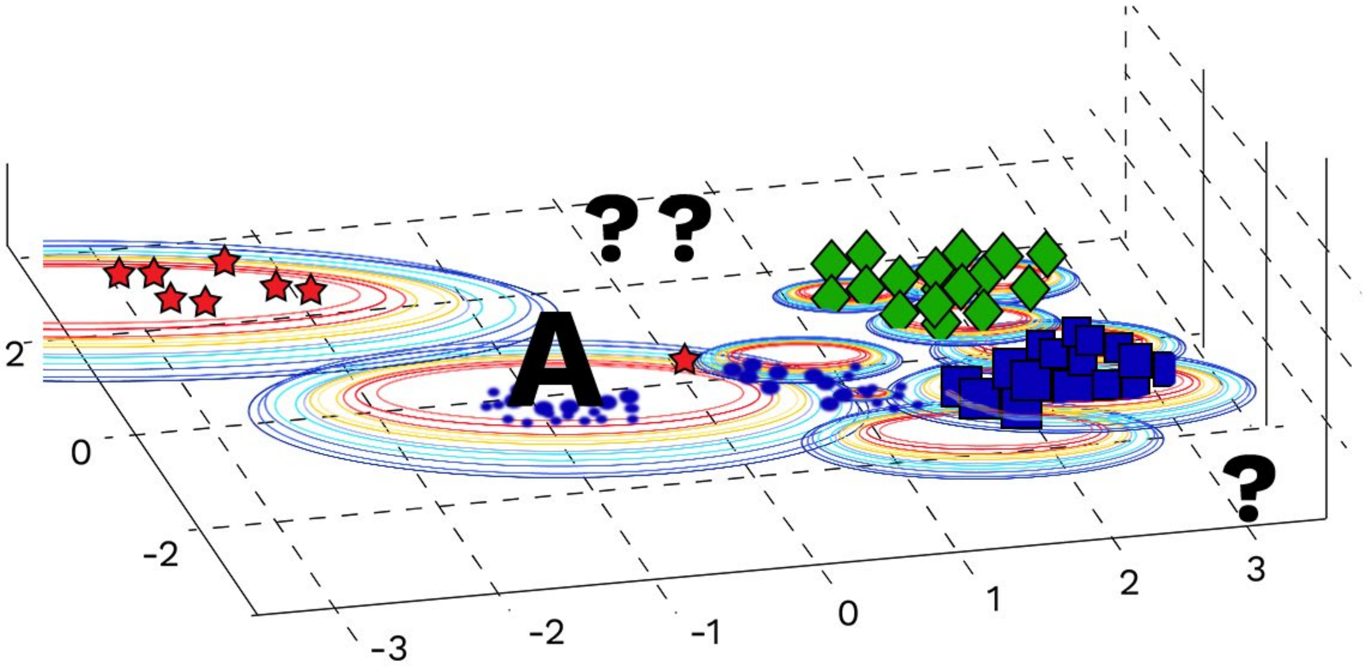
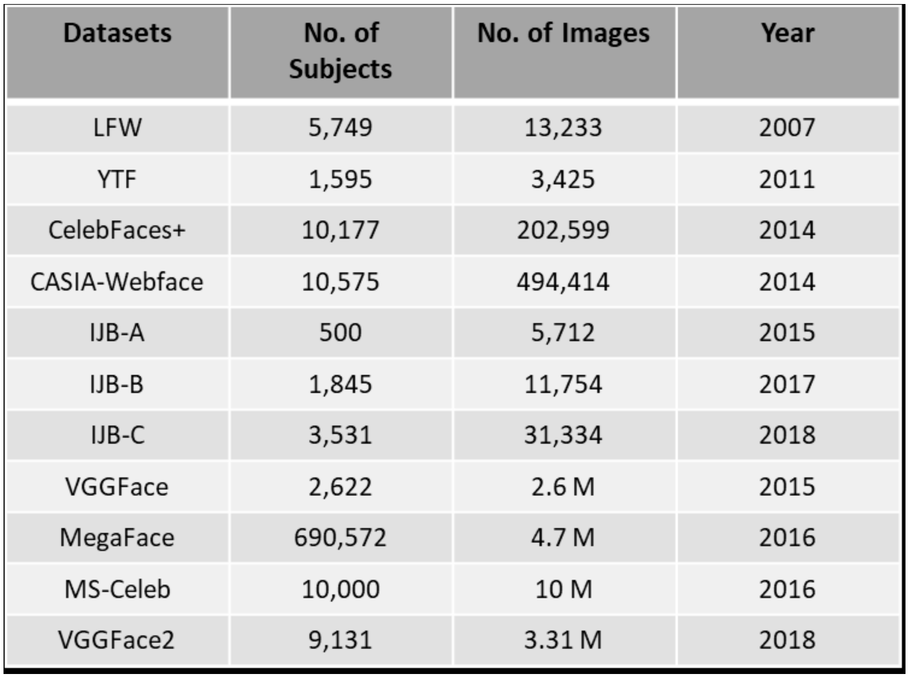


Figure 6: Showing EVM model separating the classes using circular boundary. The stars, dots, diamonds and square represent known classes and the " ?" denote the unknown classes [4]

Datasets

Table 1, shows the list of various datasets used by the face recognition community for training and testing the models. Though the datasets are open source, many cannot be used for commercial applications. We have used two different datasets, which are licensed for commercial usage, as shown in table 2. Labeled faces in wild (LFW) [6] is one of the widely used datasets in the face recognition community. It has 5749 classes among that 1640 classes has more than two images, on a total contributing 13233 images. VGGFace2 [7] dataset has more than 3 million images with 9131 classes.



Datasets	No. of Subjects	No. of Images	Year
LFW	5,749	13,233	2007
YTF	1,595	3,425	2011
CelebFaces+	10,177	202,599	2014
CASIA-Webface	10,575	494,414	2014
IJB-A	500	5,712	2015
IJB-B	1,845	11,754	2017
IJB-C	3,531	31,334	2018
VGGFace	2,622	2.6 M	2015
MegaFace	690,572	4.7 M	2016
MS-Celeb	10,000	10 M	2016
VGGFace2	9,131	3.31 M	2018

Table 1: List of available opensource datasets for training face recognition models [4]

	Labeled faces in wild (LFW)	VGG Face 2
Number of classes	5749	500
Number of images	13233	3.3 million
Images per class	1640 Classes with more than 2 images	87 to 540
License	Apache 2.0	Creative common attribution 4.0

Table 2: Details of the datasets used for training and testing our face recognition model [6] & [7]

Accuracy obtained on the datasets

The face recognition model was trained with both datasets mentioned in table 2. Training the model with the entire VGGface2 dataset is computationally expensive and time consuming. Also, as we are focused on biometric applications, 3.3 million images are not necessary for training. Therefore, the model is initially trained with the VGGFace2 dataset with 200 classes and 150000 images for training and 4000 images for testing as shown in table 3. We were able to achieve an average accuracy of 96% with 93% in known classes and 99% in unknown classes. The model was trained and tested with an LFW dataset with 1670 classes and 9000 images. We achieved an average accuracy of 95.7% with 92.7% on known classes and 99% on unknown classes.

Dataset	Classes	Training images	Testing images	Accuracy %		Average accuracy %
				Known	Unknown	
VGG Face2	200	150000	4000	93	99	96
LFW	1670	~7500	~1500	92.70	99	95.7

Table 3: Accuracy of face recognition model obtained in LFW and VGGFace2 dataset

Optimization technique

The CNN models used in MTCNN and Facenet are optimized using TensorRT packages available with the Nvidia platform. Optimization of the CNN models focuses on reducing the model size and inference time. As shown in figure 7, TensorRT uses precision calibration, layer and tensor fusion, kernel auto-tuning, dynamic tensor memory, and multi-stream execution for optimizing the model. Precision calibration changes the precision of the model from FP32 to FP16 or from FP16 to UINT8. This helps to reduce the size of the model, which in turn reduces the model loading time. Layer and tensor fusion combine similar mathematical operations in the neural network to reduce the inference time. Multi-Stream execution helps to run the algorithm in parallel with the target hardware. Applying all these techniques, the CNN model is converted to an engine file and executed in the Jetson Xavier AGX board.

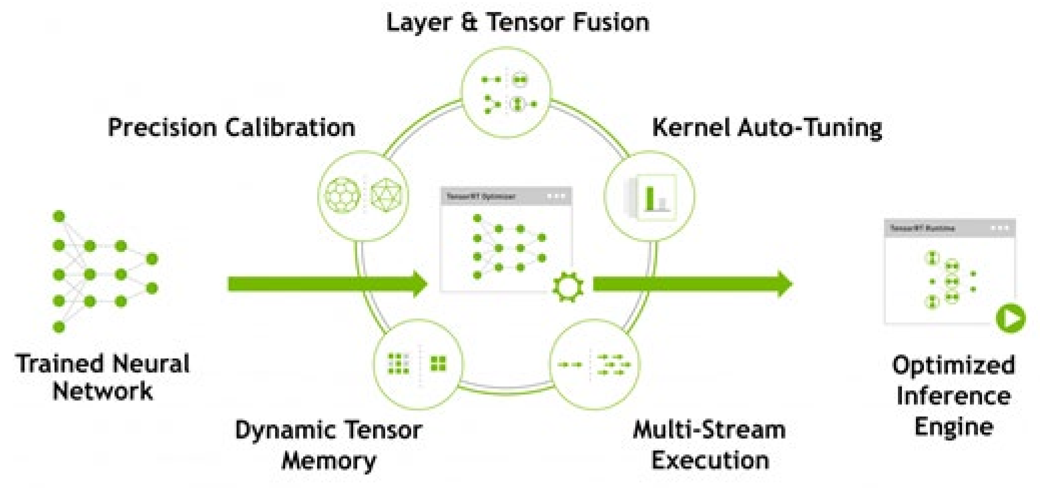


Figure 7: Different optimization techniques used by TensorRT [8]

Results after optimization

The optimized MTCNN and the facenet model are implemented in Jetson Xavier AGX. Table 4, shows the comparison of throughput for the facenet model. Raw un-optimized input model gives a throughput of 100 ms, which reduces 10 times on using Tensorflow-TensorRT (optimization package that comes with TensorFlow). While optimizing the model with the TensorRT package alone, the throughput is around 4 ms, which is 25 times less compared to the un-optimized model.

Facenet	Throughput
Tensorflow - TensorRT	10 ms
TensorRT only	4 ms
Keras - No optimization	100 ms

Table 4: Results comparing the inference time of facenet model before and after optimization

Table 5, shows the comparison of throughput for the MTCNN model. MTCNN is a Caffe based model, so it could not be optimized using the Tensorflow-TensorRT package. Therefore, the model is optimized using the TensorRT package only. The throughput of the un-optimized model was around 180 ms, which reduces to 15 ms after optimization with the TensorRT package. All the throughput values in table 4 and table 5 are checked with an image of resolution of 1980 x 1080.

MTCNN	Throughput
TensorRT only	15 ms
Caffe - No optimization	~ 180 ms

Table 5: Results comparing the inference time of MTCNN model before and after optimization

Table 6, shows the inference speed in frames per second (FPS) on the Xavier AGX board while using the different resolution of images. The speed reduces as the resolution increases, a maximum frame rate of 25 achieved with resolution 720 x 406, and the frame rate drops to 12 using an image of resolution 1980 x 1080.

Resolution	FPS (frames/s)
702 x 406	25
1280 x 720	18
1980 x 1080	12

Table 6: Results showing the inference speed in FPS using different resolution images

One shot learning

To check the performance of the model with one shot learning, the model was trained with four classes with a single image on each class. Table 7, shows the different classes used with few variations on the face. Class 'Employee 1' was trained with spectacles and the model was able to predict accurately even without spectacles as shown in figure 8. In class 'Employee 2', the model detects the mask and notifies to remove the mask. After removal, the model identifies the class successfully as shown in figure 9. In class 'Employee 3', the model was able to learn enough features though the face was covered with beard and hat, as shown in figure 10. Figure 11 shows the successful prediction of class 'Employee 4' at 45° face rotation.

Class names	Variations
Employee 1	With/without spectacles
Employee 2	With/without mask
Employee 3	Hat and beard
Employee 4	45° face rotation

Table 7: Different classes used in one shot learning with variations in faces

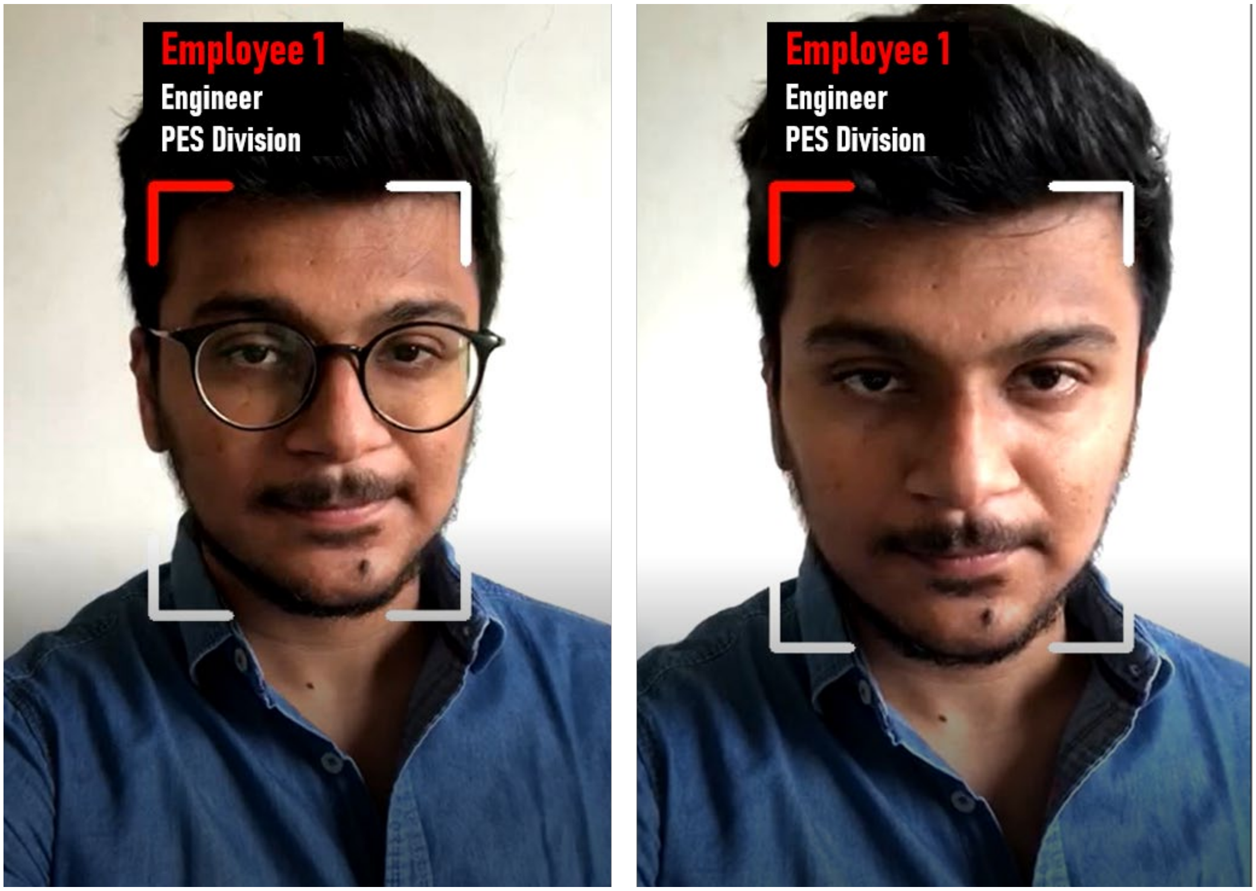


Figure 8: Recognizing face with and without spectacles

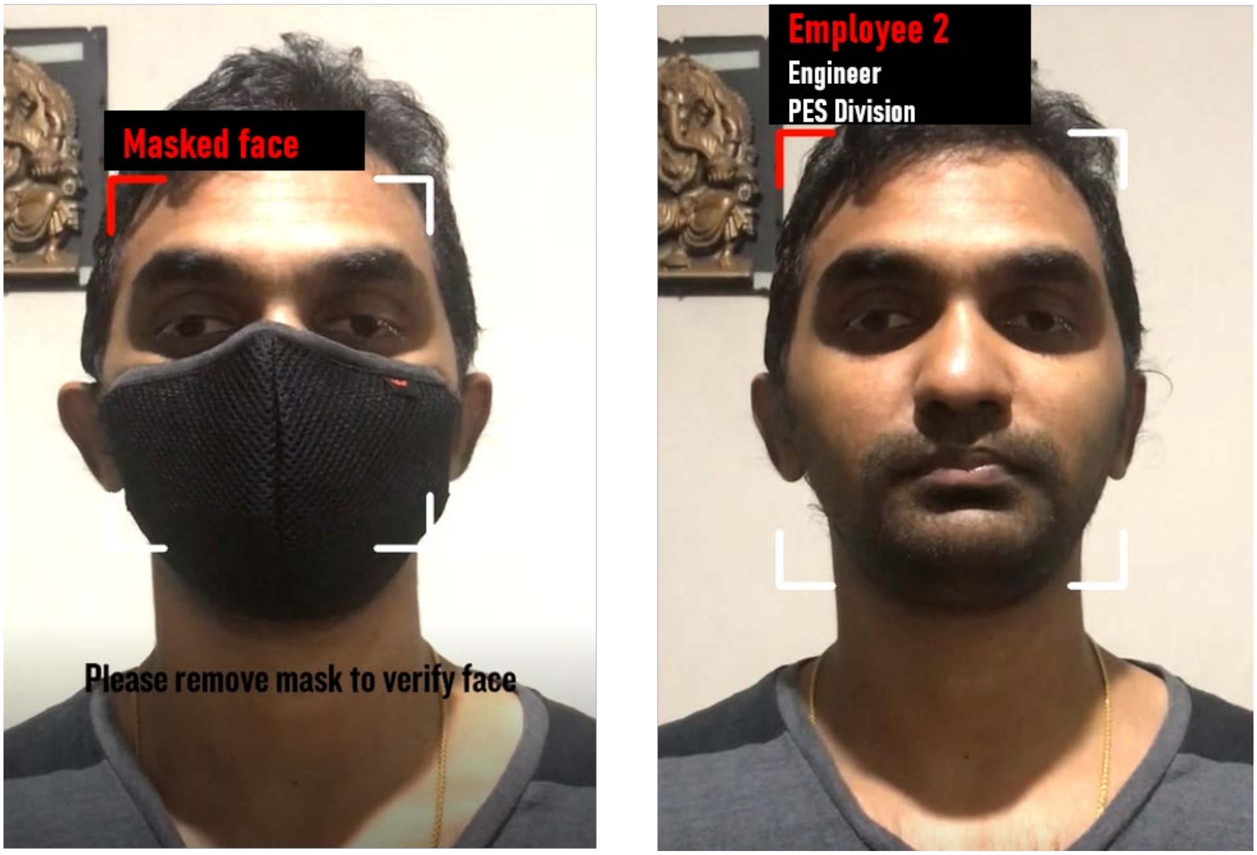


Figure 9: Face recognition with and without wearing mask

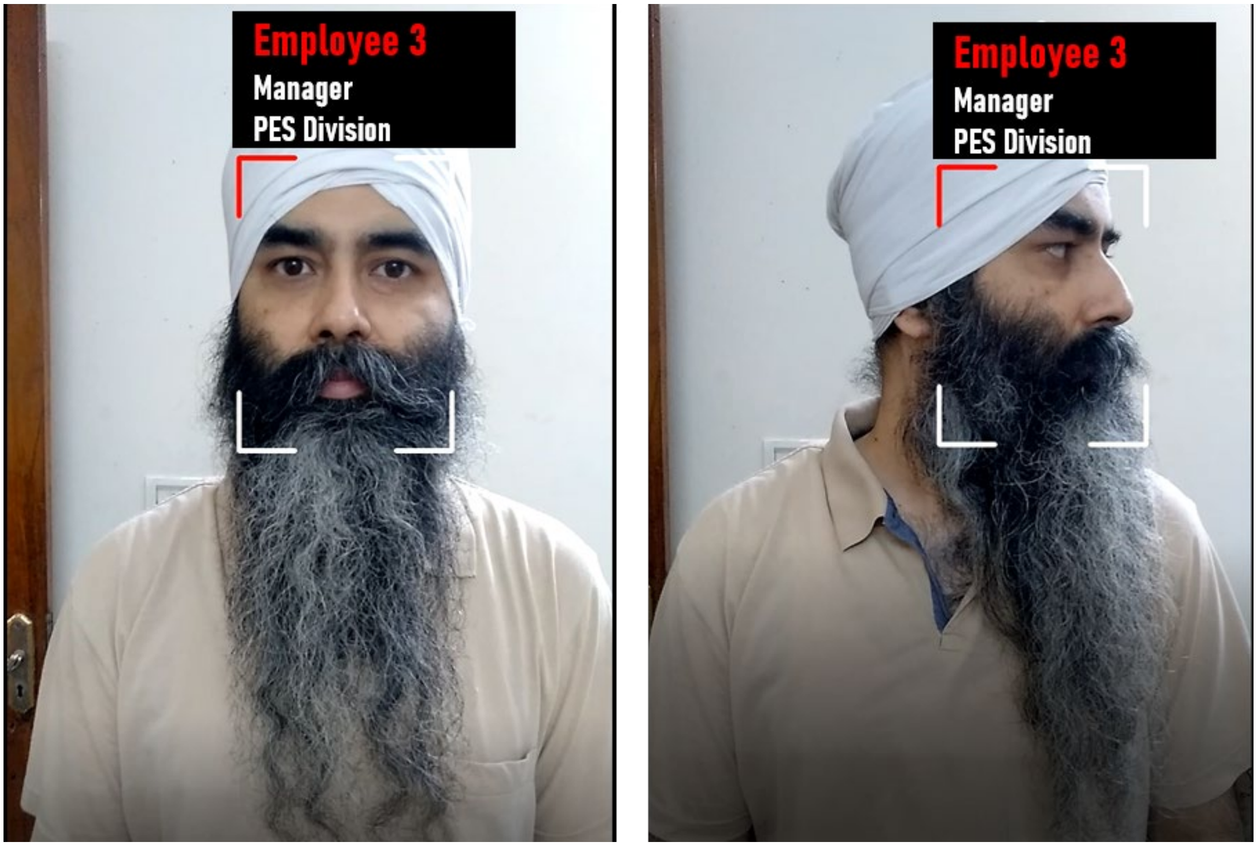


Figure 10: face recognition with hat and beard

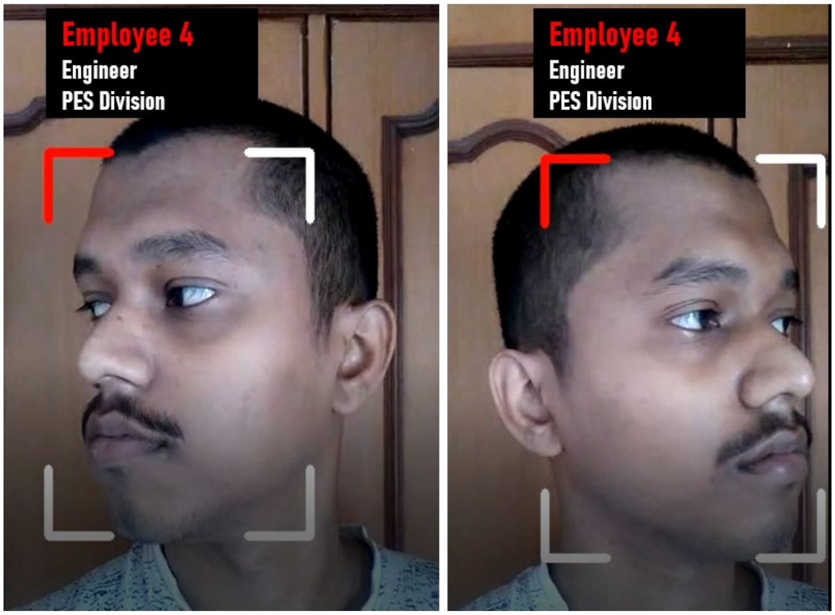


Figure 11: Recognizing faces with 45-degree side angles

Limitations

The face recognition model developed has few limitations for usage as below,

- The model is developed for applications where the face and camera are in the range of 2 to 3 feet.
- One shot learning is tested using images with clear face visibility
- The minimum image resolution tested is 720 x 406

Conclusions

The face recognition model developed and deployed on the Nvidia Jetson Xavier AGX board can achieve an average accuracy of 96% with well-known datasets like LFW and VGGFace2. The model is optimized using the TensorRT module, which improved the inference speed 10-15 times. The model performs well in one shot learning, handling variations like masks, spectacles, etc. The accuracy of the model can further be improved from 96% to 99% to achieve a state of art results.

About author

Somasundaram S., works as a Data scientist at Einfochips, an Arrow Company. He has more than 6 years of experience in various domains like Digital Image processing, Machine learning and Deep learning techniques. He holds a Master of Science degree from Indian Institute of Technology, Madras.

References:

1. https://www.mordorintelligence.com/industry-reports/facial-recognition-market
2. https://www.techsciresearch.com/report/global-facial-recognition-market/1585.html
3. K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," in IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1499-1503, Oct. 2016, doi: 10.1109/LSP.2016.2603342.
4. F. Schroff, D. Kalenichenko, J. Philbin, "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)", 2015, pp. 815-823
5. E. M. Rudd, L. P. Jain, W. J. Scheirer and T. E. Boult, "The Extreme Value Machine," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 3, pp. 762-768, 1 March 2018, doi: 10.1109/TPAMI.2017.2707495.
6. https://www.tensorflow.org/datasets/catalog/lfw
7. http://www.robots.ox.ac.uk/~vgg/data/vgg_face2/
8. https://developer.nvidia.com/tensorrt