How to Cite:
Deshmukh, R., Prajapati, A., Jain, V., & Chothale, P. (2022). Centrally controlled on-chip criminal face recognition embedded in traffic cameras. *International Journal of Health Sciences*, *6*(S6), 3864–3873. https://doi.org/10.53730/ijhs.v6nS6.10145

Centrally controlled on-chip criminal face recognition embedded in traffic cameras

Dr. Rutuja Deshmukh
Department of Electronics and Telecommunication, D.Y. Patil College of Engineering Akurdi, Pune – 411044
*Corresponding author email: rdeshmukh@dypcoeakurdi.ac.in

Abhay Prajapati
Department of Electronics and Telecommunication, D.Y. Patil College of Engineering Akurdi, Pune – 411044

Vedant Jain
Department of Electronics and Telecommunication, D.Y. Patil College of Engineering Akurdi, Pune – 411044

Pratik Chothale
Department of Electronics and Telecommunication, D.Y. Patil College of Engineering Akurdi, Pune – 411044

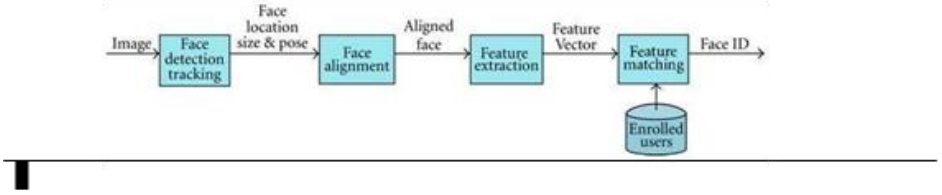
***Abstract*---**This paper evaluates the ability of convolutional networks to solve the problems arising with face classification in a constrained environment. It has the design and implementation of Siamese architecture used for face verification using a single set of the photograph. Because of the intrinsic nature of the problem, computer vision is not only a computer science area of research, but also the object of neuroscientific and psychological studies, mainly because of the general opinion that advances in computer image processing and understanding research will provide insights into how our brain works and vice versa. In the scope of the paper, the training process is closely monitored and we evaluate several practices and parameters as well as their impact on network learning. This paper introduces some novel models for all steps of a face recognition system using embedded computers. In the step of single-shot face recognition, we propose a hybrid model combining Openface artificial neural network combined with siamese network architecture to solve the process efficiently. In the first step, faces detected by Google's mediapipe library will be aligned by Geometric corrections. In this alignment step, we propose a new 2D local texture model based on relative eyes inclination. This geometric correction significantly improves the accuracy and the robustness of local searching on faces with

expression variation and ambiguous contours. In the feature extraction step, we describe a methodology to use openface neural network to extract the features and convert them into a numpy array with 128 feature values. In the face matching step, we apply a model combining two such openface networks using siamese architecture which returns a similarity score for the two feature arrays. The model links many Neural Networks together, so we call it Multi Artificial Neural Network.

***Keywords*---**face recognition, machine learning, open face

Introduction

Face recognition is a challenge of visual pattern recognition. In more detail, a face recognition system with an arbitrary image input will search a database to identify people in the input image. As shown in Figure 1, a face recognition system consists of four modules: detection, alignment, feature extraction, and matching, with localization and normalization (face detection and alignment) serving as processing steps preceding face recognition (facial feature extraction and matching).



This paper is an attempt to locate wanted targets based on their facial features and wide range reconnaissance to pick apart faces in crowds. It implements the idea in two modules using siamese architecture. First module contains a micro-controller such as Raspberry Pi housed in traffic and security cameras which will run a tflite model for real time face detection and extraction of the 256 dimensional embeddings of the available faces. Second module is a central unit which contains the database image of the target faces. This unit will compute the similar 256 dimensional embedding. These embeddings will then be shared over to the first modules in a range of networks which will then be compared with the available faces over cosine similarity.

Convolutional Neural Networks

A convolutional neural network (CNN, or ConvNet) is a type of artificial neural network (ANN) used to evaluate visual imagery in deep learning. CNNs (Shift Invariant or Space Invariant Artificial Neural Networks) are built on the shared-weight architecture of convolution kernels or filters that slide along input features and create translation-equivariant outputs known as feature maps. In comparison to other image classification methods, CNNs require very little pre-processing. This means that the network learns to optimize the filters (or kernels) by automatic learning, as opposed to hand-engineered filters in traditional techniques. This

lack of reliance on prior information or human intervention in feature extraction is a significant benefit.

Siamese Architecture

- What Are Siamese Networks?
  A siamese neural network (SNN) is a type of neural network design in which two or more sub-networks are identical. "Identical" in this context refers to the fact that they have the identical setup, including the same parameters and weights.
- What is the function of a Siamese network?
  The Siamese network's goal is to distinguish between the two inputs X1 and X2. The final layers of both networks are then sent into a contrastive loss function, which determines how similar the two images are. The network's output is a probability between 0 and 1, with a value closer to 0 indicating a prediction of dissimilar images and a value closer to 1 indicating a prediction of comparable images.

Tensor Flow Lite model

A TensorFlow Lite model is represented in a special efficient portable format known as FlatBuffers (identified by the .tflite file extension). This provides several advantages over TensorFlow's protocol buffer model format such as reduced size (small code footprint) and faster inference (data is directly accessed without an extra parsing/unpacking step) that enables TensorFlow Lite to execute efficiently on devices with limited compute and memory resources.

Embeddings

An embedding is a low-dimensional vector representation that captures relationships in higher dimensional face data. Distances between embedding vectors capture similarity between different data points, and can capture essential concepts in the original face while keeping a low memory footprint and faster interpretation.

Materials and methods

Materials
Hardware

- Raspberry Pi
- Camera Module
- A computer

Software

- Rasbian OS
- Ubuntu OS
- VS Code
- OpenFace Neural network

- Tensorflow image preprocessing library
- Tensorflow Lite
- Wifi
- sqlite3 database
- django library

Methodology

Use IOT with ML to create a product which is best of both worlds

- With the advent of technology and introduction of tensorflow lite and tensorflow micro infrastructure, We're now able to run deep learning models on small devices such as mobile phones and microcontroller.
- Using such technology in the areas of national security and surveillance can significantly improve the efficiency and reduce the footwork of the organization.
- This paper is able to run face recognition inferences on a raspberry pi device with low hardware and spatial requirements and send the signal to the controlling computer if an identity is spotted and recognized.

Fig: Example image of continuous monitoring and alert signals in case of successful detection.

Wifi/ethernet connected microcontrollers to run and update a "small" database containing limited face data for quick comparison

- Most important aspect of this paper to be considered successful is to make quick inferences.
- Keeping this in mind, sqlite3 database is used as it is fast, lightweight and reliable.
- Apart from that, all the face embeddings will be loaded into memory.
- Hence, it is important to keep only a limited amount of face data on the remote module(raspberry pi) and consequently, to strategically decide where to place each face-data so as to increase the efficiency of the whole network.

Use OpenFace in a siamese architecture to run real time face recognition on live feed

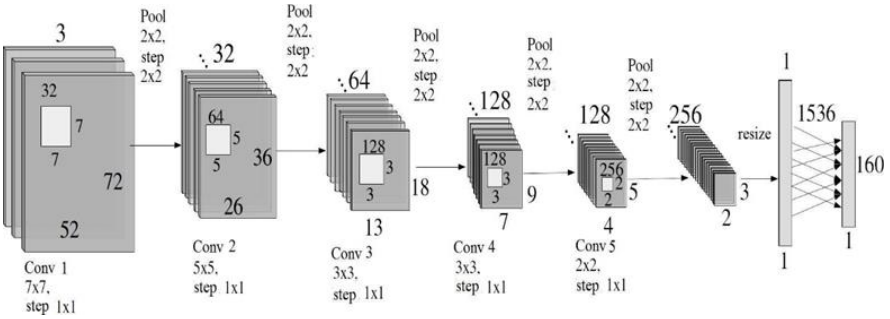


Fig: A sample image explaining the neural network and feature extraction

Fig: Complete Architecture of OpenFace Neural Network

OpenFace is an open-source state-of-the-art convolutional neural network which takes in a 96x96 color image and passes it through 166 layers of neurons to output a 128-bit vector representing the image.

Preprocessing steps for openface

- Capture the image with the required face, this can be done in two scenarios:
  - Traffic Camera
  - Upload the image to add it to database
- Detect the required face, in this case, using Google's mediapipe library.
- Crop and align the faces using facial landmarks available with mediapipe face api.

# for cropping the face x=int(detection.location_data.relative_bounding_box.xmin * image.shape[1]) y = int(detection.location_data.relative_bounding_box.ymin * image.shape[0])
h = int(detection.location_data.relative_bounding_box.height * image.shape[0]) w = int(detection.location_data.relative_bounding_box.width * image.shape[1])
face = image[y:y+h,x:x+w]

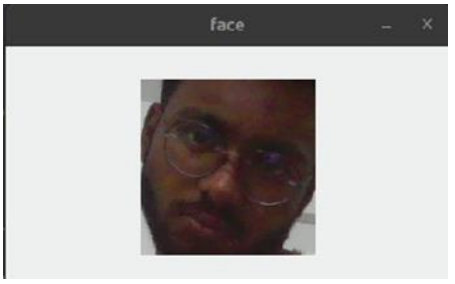
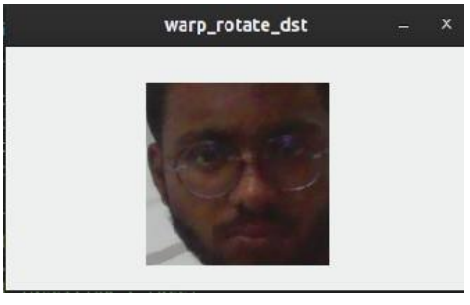


Fig: Cropped image with inclined face# for aligning the face

right_eye = detection.location_data.relative_keypoints[0] left_eye = detection.location_data.relative_keypoints[1] # calculate the angle between the eyes
dy = left_eye.y - right_eye.ydx = left_eye.x - right_eye.x
angle = np.rad2deg(np.arctan2(dy, dx))
center = (x+w//2, int(left_eye.y*image.shape[0]+ right_eye.y*image.shape[0])//2)
#calculated the center of the image and angular displacement to be corrected
rotation_matrix = cv2.getRotationMatrix2D(center, angle, 1)
warp_rotate_dst = cv2.warpAffine(image, rotation_matrix, (image.shape[1], image.shape[0]))



#correcting and warping the image to get the 2-D aligned image

After cropping and alignment, the image is ready to be sent to the neural network. However, the above steps were executed using OpenCV library, and we found that this neural network performed better when the image was read using tensorflow api. Hence, this processed image was written and loaded again into the memory using tensorflow api.

Post OpenFace

After obtaining embeddings

- At Control Center:
- These are stored in the database along with the complete profile of the person.
- These embeddings are strategically distributed across the geographical location.
- The remote modules are then monitored for the signal in case a profile is detected.

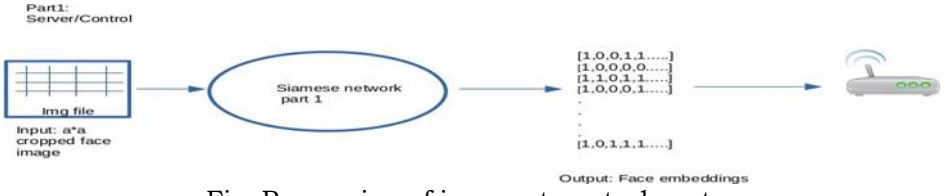


Fig: Processing of image at control center

At Remote Module

Now that we have embeddings from both camera stream and database, both of these are passed through the final stage of the siamese architecture neural network, where they are compared to give a final rating on the similarity of both the images.

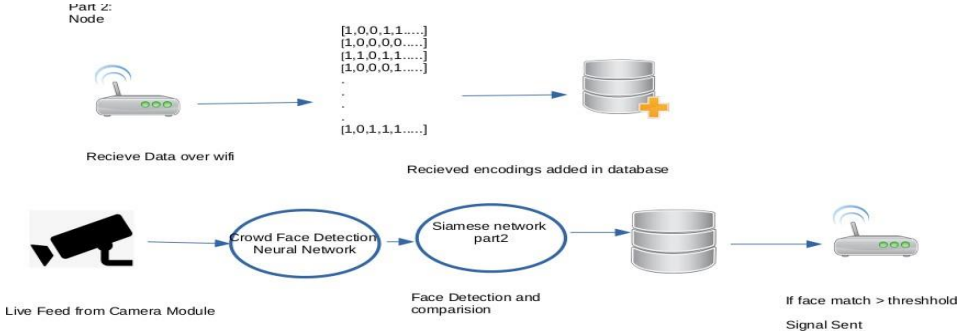


Fig: Remote Module Block Diagram with image processing steps

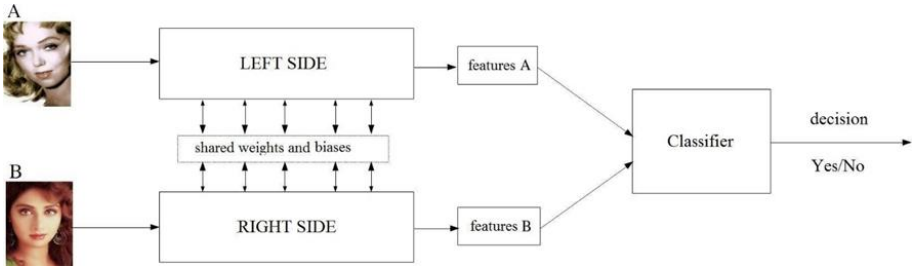
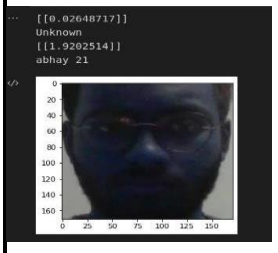
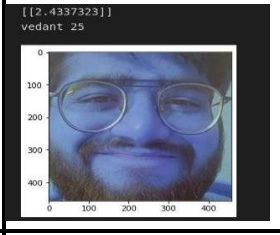
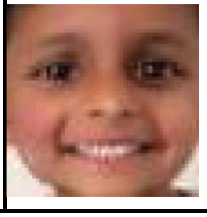


Fig. Siamese Network Architecture with example

Use the current smart traffic cams network, already existing in India, to house and support these microcontrollers to get optimal network utility

India has already started rolling out advanced traffic cameras and We've seen challan arrive directly at home. Adding face recognition to this existing network will save on a lot of infrastructure and planning.

Results and Discussion

Sr. No.	Database image	Camera Input	Screenshot	Explanation
1.				The image in the camera input is low on brightness and saturation and it is even difficult to identify with our naked eye but the neural network does this very efficiently and is able to detect "Abhay".
2.				Facenet or this neural network identifies both well lit images and dark images with the same accuracy.
3.				The two images in this network are 10 years apart and were taken before and after he hit puberty. Still the network was able to identify "Pratik".

Applications

- Prevent Retail Crime – Face recognition is extensively used to promptly find known shoplifters, organized retail criminals, or people with a history of fraud entering retail establishments.
- Finding a missing person – Facial recognition techniques like one-shot learning in CCTV systems can aid operators' efforts by enabling them to add a photo of a missing person and match it with past appearances of that face captured on CCTV system.
- Protect law enforcement – Face recognition can be used in CCTV for surveillance by law enforcement. It can be used by police to track and find past criminals suspected of perpetrating.

Conclusion

This paper presents a novel system to make face recognition available on small devices, i.e. through siamese architectured neural networks. For face detection, cropping and alignment, mediapipe library was used. It was observed that mediapipe is lightweight and produces robust results with low memory footprint and easy implementation. In the next step, OpenFace neural network is used which is an open source face recognition network consisting of more than 3 million parameters to process. However it was observed that converting OpenFace to tflite flatbuffer format caused the accuracy to slightly decrease in exchange to a significant improvement in latency and memory usage. Finally it was observed in the experiments that most of the latency was caused by the OpenFace network during the whole execution and other parts took less to negligible time as compared to it.

Future Scope

For the future scope, improvements are required in the main feature extraction function, where currently OpenFace is used. Better network will further improve accuracy and latency of the system. Furthermore, this system processes each face serially and then it is compared with each face in the database one by one. Hence, parallel computing and computational techniques can be employed to further improve on latency, thus providing real time face recognition in actual sense.

References

1. Kim, D.; Hernandez, M.; Choi, J.; Medioni, G. Deep 3D face identification. IEEE Int. Jt. Conf. Biom. 2018, 2018, 133–142.
2. Adjabi, I.; Ouahabi, A.; Benzaoui, A.; Taleb-Ahmed, A. Past, present, and future of face recognition: A review. Electronics 2020, 9, 1188.
3. M. Oravec, D. Sopiak, V. Jirka, J. Pavlovicová, and M. Budiak, "Clustering algorithms for face recognition based on client-server architecture," in 2015 International Conference on Systems, Signals, and Image Processing (IWSSIP), pp. 241–244, Sept 2015.
4. M. Khalil-Hani and L. S. Sung, "A convolutional neural network approach for face verification," in 2014 International Conference on High Performance Computing Simulation (HPCS), pp. 707–714, July 2014.

5. J. Bromley, I. Guyon, Y. LeCun, E. Siickinger, and R. Shah, “Signature verification using a“siamese” time delay neural network,” 1994.
6. X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks.,” in Aistats, vol. 9, pp. 249–256, 2010.
7. M. Bramberger, A. Doblander, A. Maier, B. Rinner, and H. Schwabach. 2006. Distributed embedded smart cameras for surveillance applications. Computer. 39: 68-75.
8. F. Yin, D. Makris, and S. A. Velastin, “Performance evaluation of object tracking algorithms,” in Proc. IEEE Int. Workshop Perform. Eval. Tracking Surveillance, 2007, pp. 733–736.
9. S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1, pp. 539–546, IEEE, 2005.
10. Shan, B. C. Lovell, S. Chen, and A. Bigdeli. 2006. Reliable Face Recognition for Intelligent CCTV. 2006 RNSA Security Technology Conference. Canberra. 356-364.
11. Widana, I.K., Sumetri, N.W., Sutapa, I.K., Suryasa, W. (2021). Anthropometric measures for better cardiovascular and musculoskeletal health. *Computer Applications in Engineering Education*, *29*(3), 550–561. https://doi.org/10.1002/cae.22202
12. Lopez, M. M. L., Herrera, J. C. E., Figueroa, Y. G. M., & Sanchez, P. K. M. (2019). Neuroscience role in education. International Journal of Health & Medical Sciences, 3(1), 21-28. https://doi.org/10.31295/ijhms.v3n1.109