Real-Time Multiple Face Recognition using Deep Learning on Embedded GPU System

Savath Saypadith[1] and Supavadee Aramvith[2]
[1,2]Department of Electrical Engineering, Chulalongkorn University, Bangkok, Thailand
[1]E-mail: Savath.S@student.chula.ac.th
[2]E-mail: Supavadee.A@chula.ac.th

***Abstract*— Face recognition can be used in several applications such as in surveillance, identification in login system and personalized technology. Recognizing multiple face in real-time on the embedded system is very challenging due to the complexity computation of the processing. In this paper, we propose multiple face recognition framework which is implemented on the embedded GPU system. The framework contains face detection based on convolutional neural network (CNN) with face tracking and state of the art deep CNN face recognition algorithm. We implemented the proposed framework into the embedded GPU system, i,e., NVIDIA Jetson TX2 board. Experimental results showed that the proposed system can recognize multiple faces up to 8 faces at the same time in real time with up to 0.23 seconds of processing time and with the minimum recognition rate above 83.67%.**

I. INTRODUCTION

Since the deep learning has dramatically drawn attention in the computer vision community, face recognition has been one of the most extensively studied topics to come up with the challenging of the recognition problem. Face recognition has used in several applications in both private and public sectors such as surveillance system, person authentication, etc. Conventional face recognition method aims to develop the faster algorithm and more robust. Whereas, accurate recognition depends on high resolution of face image and with no occlusion. Good face image should be discriminative to the change of face identity while remains robust to the intra-personal variation. Although face recognition algorithms have reached the accuracy level under certain condition, the face recognition algorithm still affected by the external and internal variation such as the illumination, pose, facial expression and occlusion. The key challenge of face recognition is to develop effective feature representation to improve the accuracy in the different scenarios. Recently, deep learning has achieved great success on computer vision research with significant improving the state-of-art in classification and recognition problems.

Face recognition algorithm based on deep learning has been proposed by many researchers in the literature which achieved a good performance in term of the processing time and accuracy [1-3]. However, multiple face recognition in real-time processing is still a problem for deep learning algorithm due to high computational complexity. In a simple task of face recognition algorithm, the raw face image has been used in the neural network to learn the face feature by using Convolutional Neural Networks (CNN), pooling and fully connected layer. These lead to the demands of computational resources of each step.

In this paper, we proposed a framework for multiple face recognition which is implemented into the embedded GPU system to recognize multiple face in real-time. The model was trained by a network architecture that reduced the network parameter, and the tracking technique was added to a framework to reduce the processing time while keeping the acceptable recognition rate.

The remainder of this paper is organized as follows. Section II reviews related work on face detection, recognition and tracking. Our proposed framework describes in Section III. Experimental and results present in Section IV. Finally, Section V concludes this paper.

II. RELATED WORKS

In this section, we firstly describe typical algorithm in the framework. The most important part in the framework can be divided into three parts: face detection and alignment, face recognition and tracking algorithm, as shown in figure 1. The literature reviews of each part describe as follows.

A. Face Detection and Alignment

An input image consists of one or many faces in a different location. To determine a face in a certain image, we need to define the general structure of a human face such as eyes, nose, forehead, chin and mouth. Later it is possible to determine the number of face, the exact location and the size of all the faces.

Viola and Jones [4] proposed the algorithm for face detection. The algorithm is scan a sub-window capable of detecting faces across a given input image. The Viola-Jones face detector contains three main ideas that can run in real time: the image integral, classifier learning with AdaBoost, and the attentional cascade structure. The approach would be rescaled the image to different size. Finally, the algorithm run the fixed size detector through these images. However, it may consume much time due to the calculation of the different size images.

Histogram of Oriented Gradients (HOG) feature descriptor [5] has been proposed to detect an objects in the image. The algorithm divided image into small sub-images call “cells”, then accumulating a histogram of edge orientations within that cell. The algorithm counts occurrences of gradient

orientation in localized portion of an image. Finally, HOG descriptor used Support Vector Machine (SVM) as a classifier to classify whether the face or non-face.

Some researches proved that neural network can improve the performance of face detection, but it increases the execution time. Farfade et al. [6] proposed a method called Deep Dense Face Detector (DDFD) which is able to detect faces in a wide range of orientations using a single model based on deep convolutional neural networks. The method dose not required segmentation, bounding-box regression, or SVM classifiers. The study found that the method is able to detect faces from different angles and there seem to be correlation between distribution of positive examples in the training set and score of the proposed face detector.

Zhang et al. [7] proposed a framework to detect face and alignment using unified cascaded Convolutional Neural Networks (CNN) by multi-task learning. The framework consists of three stages: first, to obtain the bounding box regression, utilizing a convolutional network called Proposal Network (P-Net). After getting the estimated bounding box regression candidates, then employ non-maximum suppression (NMS) to ignore bounding boxes that significantly overlap each other. All candidates are fed to Refine Network (R-Net), R-Net aims to reject a large number of false candidates, performs calibration with bounding box regression, and NMS candidate merge. Finally, the Output Network (O-Net) describe the details of the face and output five facial landmarks' position. On a 2.60 GHz CPU, it is possible to achieve a framerate of 16 frames per second (fps). With Nvidia Titan Black GPU, the achievable frame rate can be up to 99 fps. The approach achieved 95% in average of accuracy in each stage across several challenging benchmarks while keeping real-time performance.

B. Face Recognition

The most important part of face recognition system is to learn the unique feature of the face to be able to differentiate distinct faces. It is important to design face recognition algorithm to be resilient under ill-illuminated conditions. Previously, conventional face recognition approaches based on geometric features [8], principle component analysis (PCA) [9] and linear discriminant analysis [10] may achieve great recognition rate in a certain condition. Recently, face recognition based on CNN have been proposed with algorithms higher accuracy. We briefly reviewed some of state of the art methods as follows.

Taigman et al. [3] proposed an algorithm for face recognition called DeepFace which is combined 3D alignment and similarity metric with deep neural network. The network architecture is learned from the raw pixel RGB values. 67 fiducial points are extracted by a Support Vector Regressor (SVR) trained to predict point configurations from an image descriptor based on Local Binary Pattern (LBP) histogram. The DeepFace runs at 0.33 seconds per image on single core Intel 2.2 GHz CPU. As a result, the system achieved 97.35% in accuracy. Sun et al. [2] proposed the method called DeepID3. The method proposed two deep network architectures based on stacked convolution and inception layers to make them suitable to face recognition. By improving the method from DeepID2+, DeepFace utilized Joint identification-verification supervisory signals to the last fully connected layer as well as a few fully connected layers branches out from pooling layer. This method is also used in DeepID3 to make the architecture better supervise early feature extraction processes. In addition, the DeepID3 network is significant deeper compared to DeepID2+ by using ten to fifteen non-linear feature extraction layers. As a result, DeepID3 archived 99.53% for face verification accuracy and 96.0% for face identification accuracy.

The state-of-art face recognition has been proposed by Schroff et al. [1] called FaceNet which based on Deep convolutional network. The network is trained to directly optimize the embedding of the input image. The approach is directly learning from the pixels of the face image. FaceNet used the triplet loss to minimize the distance of the embedding in the feature space between an anchor (reference face) and a positive (the same person) and maximize the anchor and negative (a different person). it is considered the face of the same person have small distances, where the faces of distinct person have large distance. The network is trained such that the squared L2-norm distances in the embedding space directly correspond to face similarity. FaceNet trained the CNN using Stochastic Gradient Descent (SGD) with standard back propagation and AdaGrad. The approach achieved the accuracy of 99.63%. In addition, the system used 128 bytes per face which is suitable for running in the embedded computer vision.

C. Tracking

Both face detection and recognition processes can consume significant amount of the processing time. The tracking algorithm is the essential part to reduce the processing time.

Danelljan et al. [11] proposed an approach to learn discriminative correlation filters based on a scale of pyramid representation. The method improved the performance compared to an exhaustive scale search with significant scale variation. A closely related approach, proposed by Bolme et al. [12], is based on finding an adaptive correlation filter by minimizing the output sum of squared error (MOSSE). Danelljan et al. method traines a classifier on a scale pyramid. This allows to independently estimate the target scale after the optimal translation is found. The score, y, of the output can calculate by using (1).

Given an image patch z of size MxN in a new frame, the correlation score, y, is computed as

$$y = \Im^{-1}\{\bar{H}_t Z\} \tag{1}$$

Here, capital letters denote the discrete Fourier transforms (DFT), $\Im^{-1}$ denotes the inverse DFT operator. The new target location is estimated to be at the maximum correlation score of y.

III. PROPOSED FRAMEWORK

The overview of the framework is described in Figure 1. Firstly, we apply face detection and alignment algorithm to localize the face in the scene, then use detected face as an input for the recognition step. Finally, we apply the tracking algorithm along with the recognized face.

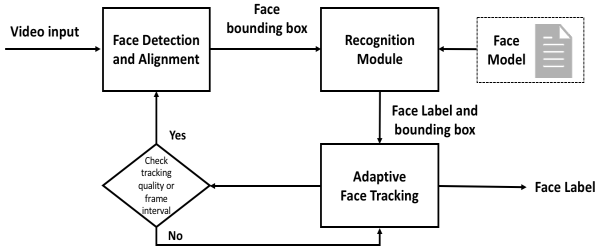


Figure 1. Overview of the proposed framework.

A. Face Detection and Alignment

Face detection is the essential part in the framework as an input to recognition step. We utilized the face detection algorithm based on [7], the algorithm based on CNN which can detect the face in many variation and illumination condition. Moreover, the output of the detection is the bounding box and facial landmark for alignment which increase the recognition accurately. Given the input image, the algorithm resizes the image into different scales which we called the image pyramid. These different images size is fed to the three stages cascaded framework: P-Net, R-Net and O-Net for face detection and alignment. The classification classifies the image whether face or non-face. Finally, the bounding box regression and facial landmark localization are extracted. All of the face image from detector is resized to 160x160 pixels before the recognition step.

B. Recognition Module

Most of the face recognition based on CNN is designed for high-performance computing due to the computational cost in the network layers. In this paper, we will implement the framework into embedded computer vision system which is used low power consumption, the architecture should be able to run in real-time without accessing to the cloud and recognize multiple face at the same time within the board.

Therefore, we applied face recognition method based on FaceNet [1], FaceNet is based on Inception architecture [13]. The goal is to minimize the embedding of the input image in the feature space of the same person while maximizing the face of different person to be far away.

To decrease parameters of the network as well as decreasing the size of the model to save the memory usage of the GPU, we utilize the SqueezeNet architecture [14] instead. This leads to the faster training and face prediction but may have a slightly drop in accuracy. Moreover, the model size also decreases. The training process, results, and parameters used in the training are described in the section IV.

C. Adaptive Face Tracking

The processing time is the main problem in face recognition system for real-time application due to the processes of each step such as face detection, face alignment and face recognition itself. In the experiments, when we applied face detection algorithm in the whole image of every frame in video, face detection consumes high processing time in each frame. We assume that when the face is recognized, face still in the scene in short period. Thus, we applied algorithm to track the face using correlation filter based on [11] algorithm.

IV. EXPERIMENTAL AND RESULTS

A. Experimental Setup

The design of hardware specification for face detection and recognition is crucial. The main challenge is the computational cost of the CNN network architecture when we implement the algorithm based on deep learning into a board that have a limit of computational resources. To overcome the problem, the board should support parallel programming which can compute multiple processes at the same time. In this experiment, we implemented a framework into NVIDIA Jetson TX2 board which is a powerful embedded board. Jetson TX2 support a 256 CUDA cores NVIDIA Pascal of GPU. It packs a high-performance processor into a small and power-efficient design that suitable for intelligence devices like robot, drones, and portable devices that rely on computer vision applications.

To measure the performance of the framework, we used Recognition Rate (RR) [15] , precision, recall, f-score [16] and computation time as metrics, RR can be calculated using (2) and the computation time of face recognition presented in seconds.

$$RR = \frac{\text{number of collect recognize}}{\text{number of total testing image}} \times 100 \quad (2)$$

The precision recall and f-measure for face detection evaluation can calculated as follows,

$$\text{Precision} = \frac{\#TP}{\#TP + \#FP} \quad (3)$$

$$\text{Recall} = \frac{\#TP}{\#TP + \#FN} \quad (4)$$

$$\text{F-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5)$$

where TP (True Positive) is a detection bounding box presents in system under test (SUT) and ground truth (GT), FP (False Positive) presents in SUT but not in GT and FN (False Negative) presents in GT but not in SUT.

B. Datasets

The dataset used in this paper can be divided into three parts: training, validation and testing where each dataset consists of the faces with different variations, illuminations, poses and occlusion. The detail of datasets is described as the following.

Figure 2. Examples of VGGFace2 [15].

To achieve highest accuracy and to reduce CNN network parameters, the dataset must contain enough samples of variation and good representation of face. If dataset has less data to train, it comes up with underfitting. In contrast, if the dataset contains lots of data, the training model become overfitting. Therefore, the training requires a good model architecture to reduce the weight of the training as well as to reach the optimal value of parameters setting such as learning rate, number of iterations, etc. The VGGFace2 dataset [17] has been used for training the network. The dataset contains 3.31 million images of 9,131 subjects. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession as shown in Figure 2.

The validation set is used for minimizing overfitting as well as tuning the parameters such as weights and biases to increase the accuracy over the dataset that has not been shown in the training before. LWF dataset [18] has been used for validate the model in this work. The dataset contains 13,233 face images with 5749 different subjects. The resolution of the images is 250x250 pixels.

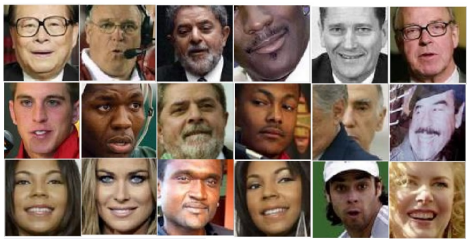


Figure 3. Examples of LWF dataset.

Test set is the face set that used to measure the performance of face recognition algorithm in term of recognition rate and time to recognize the face. YoutubeFace database [19] used to measure the performance of tracking and FEI face database [20] used for measure the recognition rate.

YoutubeFace database is a database of face videos, containing of 3425 videos of 1595 different people. We used YoutubeFace database to measure performance of tracking due the dataset contains a people movement within the scene and unconstrained of the background. Whereas, FEI face database is used to measure the recognition rate of the algorithm. The database contains 2800 images of 200 people. 14 images for one people. All the images are RGB image and rotate up to about 180 degrees in frontal position. The original size of each image is 640x480 pixels. The frontal face image and the label of each person were used to train a classifier in the experiment.

To measure the processing time of multiple face recognition, we generated the CUFace dataset by capturing a video of people standing in front of the camera with the resolution of 640x360 pixels and the frame rate of 25 fps. The distance of the people from the camera is between 1.00-1.20 meters. The example of CUFace dataset shown in Figure 4.

Figure 4. Example of CUFace dataset.

C. Model and Classifier Training

The model was trained on SqueezeNet architecture with VGGFace2, the input size of image is 224x224 pixels. The parameters used for training are initialized with the learning rate of 0.1, randomly crop and flip an image, training with 2000 epochs.

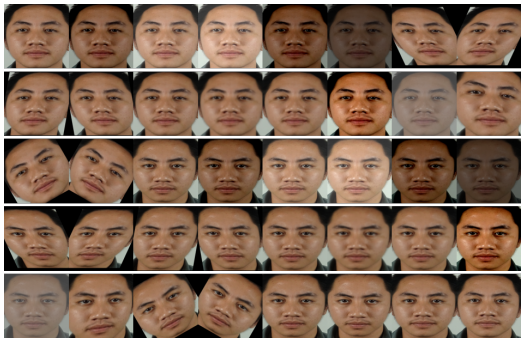


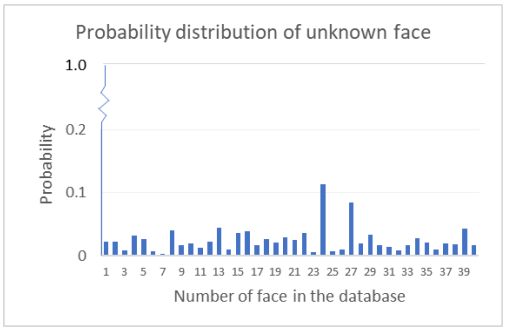
Figure 5. Example of 40 image generated from image augmentation technique.

To train a classifier, we cropped and aligned the image and resized to 160x160, then utilizing image augmentation technique to generate 40 faces (as shown in Figure 5) from original face images with many variations such as adding noise, illumination, rotatation, shear the image, etc. These 40 images are used as an input to train a classifier. The accuracy of the recognition is varied by the training sample as we will discuss in the results.

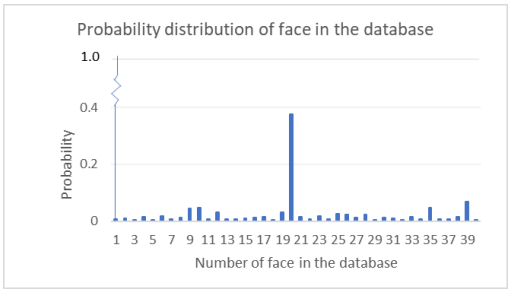
Support Vector Machine (SVM) classifier is used to classify the person in the feature space. The output of the prediction is the probability distribution of the person in the database which can be calculated by (6) based on [21]. The maximum prediction is considered as a label of the person if the value greater than the threshold value, otherwise, we considered as unknown person. SVM provide stable results and faster training while softmax might be stucked by all the calculations if we have a complex training data with many labels.

$$P(label / input) = \frac{1}{(1+\exp(\mathrm{A}* f(input)+\mathrm{B})} \quad (6)$$

Where *P(label/input)* is the probability that input belongs to label and *f(input)* is a signed distance of the embedding in the feature space, *A* and *B* are the parameters from the training classifier.



(a)



(b)

Figure 6. (a) Probability distribution of predicting unknown face. (b) Probability distribution of predicting face in the database.

In the experiment, we found that the maximum of the probability is varied by the amount of person in the database. In other word, when increasing the amount of person in the database, the maximum probability is decrease. This lead to varying the threshold when the database is increased. As we show in Figure 6, when recognizing the face in database, the maximum probability is totally higher than others compared with recognizing the unknown face.

In this experiment, we used (7) to calculate the threshold of the predicting δ by the assumption that the probability of the prediction has the ratio difference between the maximum and the others.

$$\delta = C \times \frac{\sum_{1}^{N} x_i - x_{\max}}{N-1} \quad (7)$$

where *C* is the ratio value between maximum probability and the others (*C*=1,2, …, n), x_i denotes the probability of i^{th} person, x_{max} is the maximum probability of the prediction, and *N* is the total number of persons in the database. In the experiment, the optimal value of *C* is equal to 10 which archive the highest recognition rate.

D. Experimental results

The experimental results in this paper can be divided into three parts: performance of tracking, recognition rate for multiple faces, recognition rate and processing time on CUFace dataset.

1. Tracking performance

To show the tracking algorithm performance, we tested the detection algorithm with YoutubeFace database which contains the video clip of people movement. The tracking algorithm performance is comparable to normal detection where the processing time is faster. The results are shown in Table 1.

Table 1. Comparison of average precision, recall and f-score of detection and detection with tracking.

	Precision	Recall	F-Score
Detection	0.75	0.82	0.76
Detection with tracking	0.69	0.99	0.75

2. Recognition rate for multiple faces

The recognition rate depends on the number of trained faces. We test the algorithm with FEI database. The test set consists of people face images in different direction which is difficult to recognize when the features of face does not appear. The result in Figure 1 indicates that 40 trained face images achieves the highest recognition rate.

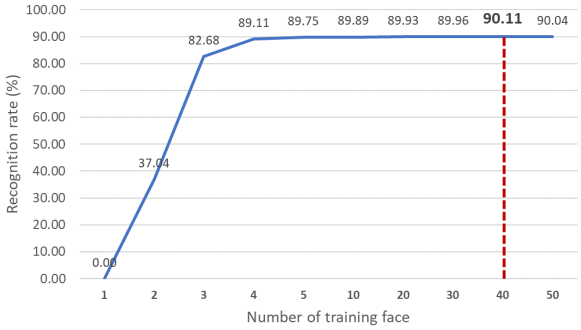


Figure 7. Comparison of recognition rate and number of training face on FEI database.

3. Recognition rate and processing time

To measure recognition rate and the processing time of recognition algorithm of multiple faces, we tested the algorithm with CUFace dataset. We compared normal recognition and recognition using tracking algorithm. The processing time is speeded up by 56.10% on average. The frame rate varied by the number of face in recognition, our algorithm can recognize multiple face around 5-10 fps with recognition rate around 90%.

Table 2. Comparison of processing time between recognition and recognition with tracking.

Number of face	Processing time (s)		Speed up (%)	RR(%) of recognition with tracking
	Recognition	Recognition with tracking		
1	0.19	0.05	75.79	96.00
2	0.23	0.07	69.78	98.67
3	0.25	0.09	63.39	95.11
4	0.28	0.12	57.30	94.50
5	0.32	0.16	50.31	84.73
6	0.35	0.20	44.13	84.11
7	0.39	0.22	43.78	83.67
8	0.41	0.23	44.31	85.50
Average	0.30	0.14	56.10	90.29

The outputs of the recognition are shown in the Figure 8. The bounding box is located the face with the label (name) of the person.

Figure 8. The output of the recognition. The bounding box located the face and showing the name of each person.

V. Conclusions

In this paper, we proposed a framework for real-time multiple face recognition. The recognition algorithm based on CNN which is the state-of-art algorithm. The framework consists of the tracking technique and using the minimal weight of the model. This can reduce the processing time and network parameter to learn recognize multiple face feature in real-time. The framework is implemented into NVIDIA Jetson TX2 board which supports CUDA, the advantage of the board is able to run the process in parallel. As a result, the processing time is faster and suitable for real-time multiple face recognition with acceptable recognition rate.

VI. Acknowledgement

This work is supported in part by AUN/SEED-Net (JICA), Collaborative Research Project entitled Video Processing and Transmission. This support is gratefully acknowledged.

References

[1] 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.

[2] Y. Sun, D. Liang, X. Wang, and X. Tang, "Deepid3: Face recognition with very deep neural networks," *arXiv preprint arXiv:1502.00873,* 2015.

[3] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701-1708.

[4] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision,* vol. 57, no. 2, pp. 137-154, 2004.

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, vol. 1, pp. 886-893: IEEE.

[6] S. S. Farfade, M. J. Saberian, and L.-J. Li, "Multi-view face detection using deep convolutional neural networks," in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, 2015, pp. 643-650: ACM.

[7] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters,* vol. 23, no. 10, pp. 1499-1503, 2016.

[8] L. Wiskott, N. Krüger, N. Kuiger, and C. Von Der Malsburg, "Face recognition by elastic bunch graph matching," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 19, no. 7, pp. 775-779, 1997.

[9] J. Yang, D. Zhang, A. F. Frangi, and J.-y. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," *IEEE transactions on pattern analysis and machine intelligence,* vol. 26, no. 1, pp. 131-137, 2004.

[10] W. Zhao, A. Krishnaswamy, R. Chellappa, D. L. Swets, and J. Weng, "Discriminant analysis of principal

components for face recognition," in *Face Recognition*: Springer, 1998, pp. 73-85.

[11] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*, 2014: BMVA Press.

[12] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 2544-2550: IEEE.

[13] C. Szegedy *et al.*, "Going deeper with convolutions," 2015: Cvpr.

[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size," *arXiv preprint arXiv:1602.07360,* 2016.

[15] B. Dai and D. Zhang, "Evaluation of face recognition techniques," in *PIAGENG 2009: Image Processing and Photonics for Agricultural Engineering*, 2009, vol. 7489, p. 74890M: International Society for Optics and Photonics.

[16] A. Baumann *et al.*, "A review and comparison of measures for automatic video surveillance systems," *EURASIP Journal on Image and Video Processing,* vol. 2008, no. 1, p. 824726, 2008.

[17] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "VGGFace2: A dataset for recognising faces across pose and age," *arXiv preprint arXiv:1710.08092,* 2017.

[18] G. B. Huang and E. Learned-Miller, "Labeled faces in the wild: Updates and new reporting procedures," *Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep,* pp. 14-003, 2014.

[19] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2011, pp. 529-534: IEEE.

[20] C. E. Thomaz and G. A. Giraldi, "A new ranking method for principal components analysis and its application to face image analysis," *Image and Vision Computing,* vol. 28, no. 6, pp. 902-913, 2010.

[21] J. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Advances in large margin classifiers,* vol. 10, no. 3, pp. 61-74, 1999.