Towards a Real-Time Facial Analysis System

Bishwo Adhikari[1] Xingyang Ni[1] Esa Rahtu[1] Heikki Huttunen[2]
[1]Tampere University, Finland [2]Visy Oy, Finland
{bishwo.adhikari, xingyang.ni, esa.rahtu}@tuni.fi heikki.huttunen@visy.fi

***Abstract*—Facial analysis is an active research area in computer vision, with many practical applications. Most of the existing studies focus on addressing one specific task and maximizing its performance. For a complete facial analysis system, one needs to solve these tasks efficiently to ensure a smooth experience. In this work, we present a system-level design of a real-time facial analysis system. With a collection of deep neural networks for object detection, classification, and regression, the system recognizes age, gender, facial expression, and facial similarity for each person that appears in the camera view. We investigate the parallelization and interplay of individual tasks. Results on common off-the-shelf architecture show that the system's accuracy is comparable to the state-of-the-art methods, and the recognition speed satisfies real-time requirements. Moreover, we propose a multitask network for jointly predicting the first three attributes, i.e., age, gender, and facial expression. Source code and trained models are available at https://github.com/mahehu/TUT-live-age-estimator.**

***Index Terms*—face detection, face recognition, facial similarity, real-time system**

I. Introduction

Human facial analysis is one of the widely studied topic in computer vision that includes face verification [1], [2], head pose estimation [3], [4], facial expression recognition [5], [6] and age estimation [7], to name a few sub-domains. While computer programs have traditionally been unable to analyze facial images, humans are very good at spotting even the smallest differences. With the surge of deep learning techniques, algorithms have surpassed human accuracy in most of the above tasks. More recently, deep learning has opened new avenues for applications of real-time facial analysis, such as facial identification [1], [2] and security surveillance [8].

In the field of facial image analysis, the majority of works focus on improving the accuracy of a specific task. Less attention is paid to investigate the computational complexity [9]–[11], in particular at the system level, where the architect needs to pay attention to the functionality of the entire system as well as that of the individual components; simultaneously optimizing for prediction accuracy, inference speed, memory footprint, parallelization as well as user experience (*i.e.*, the system should at least *appear* smooth although some components might operate at below real-time speed).

The straightforward approach would sequentially first detect all faces, then estimate their age, gender, facial expression, and facial similarity; show the result on the screen and start over with the detection. However, the refresh rate on screen would be dictated by the sum of execution times of individual components. On the other hand, users are less sensitive to a slow refresh rate of age estimates than the slow refresh rate of the display itself. Therefore, the system has to prioritize the tasks differently while maximizing the performance and minimizing idle times.

Fig. 1. Our real-time facial recognition system in action. It detects human faces on a frame captured by a webcam, recognizes age, gender, and emotion in real-time. Additionally, it shows the most similar appearing face obtained from the similarity search network.

Our system consists of a screen, a camera, and a computer, and it estimates the age, gender, and facial expression of all faces seen by the camera. In addition to these functions, the most similar-looking face from a database of celebrity faces is shown next to the detected face. Apart from serving as an illustrative example of modern human-level machine learning for the general public, the system also highlights several common aspects in real-time machine learning systems. The subtasks needed to achieve these recognition results represent a wide variety of tasks, including (a) face detection, (b) age estimation, (c) gender prediction, (d) facial expression prediction, and (e) image retrieval. Moreover, all these tasks should operate in unison, such that each task will receive enough resources from a limited pool.

Overall, we make the following contributions:

- We present a detailed system-level architecture for estimating several attributes from facial images.
- We show the real-time performance of each component of the proposed architecture and its smooth functionality even on a moderate-resourced computing platform.
- We release source code and trained models, with detailed instructions for deployment.

The structure of the rest of the paper is as follows. In Section II we describe the system level multi-threaded architecture for real-time processing. This is followed by a detailed description of individual components of the system in Section III. Next, we report the experimental setups

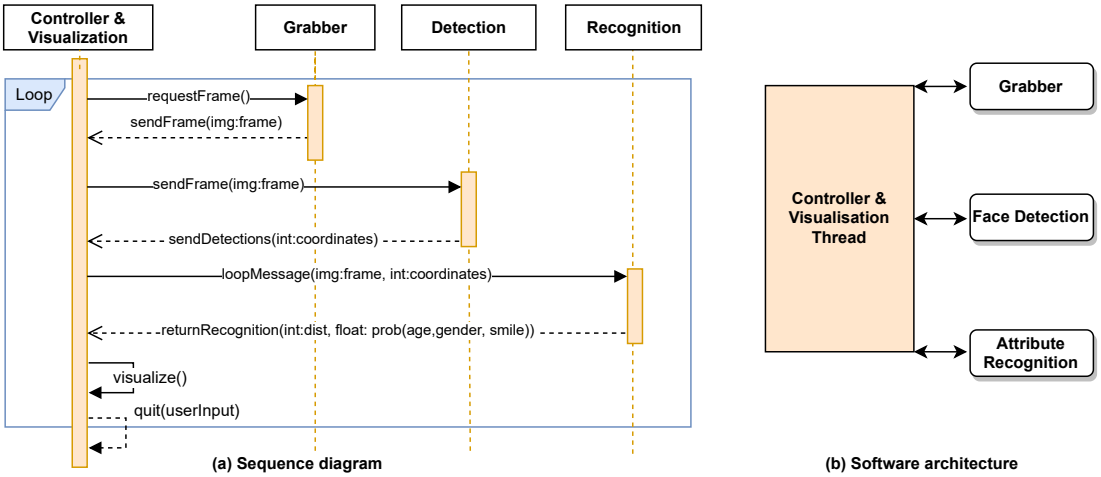

Fig. 2. Sequence diagram of the proposed real-time facial analysis system in (a) and software architecture of our system in (b).

together with datasets and performance measurement metrics in Section IV. We present experimental results of each recognition component in Section V and finally, we discuss the benefits of demonstrating the potential of modern machine learning to both the general public and experts in the field.

II. System Level Functionality

The challenge in real-time operation is that there are multiple components in the system, and each uses a different amount of execution time. The system should be designed such that the operation appears smooth, which means that the most visible tasks should be given higher priority in scheduling.

The implementation is multi-threaded, as illustrated in Fig. 2. Each thread operates asynchronously, with *recognition threads* polling for new frames to process whenever they are idle. The system is controlled by the *controller & visualization thread*, which receives new frames from the camera via the dedicated *grabber thread*. The controller thread also stores the frames in a buffer with each frame associated with flags, whether they have been processed by each of the threads. Finally, it visualizes by showing the live video as well as overlay the most recent recognition results to the user in real-time. The asynchronous threading structure also allows execution on dedicated platforms (*e.g.*, detection running on the CPU and recognition on the GPU). Also, it enables straightforward process prioritization by launching multiple recognition threads for the same task.

A. Frame Capture

The recognition process starts from the *grabber thread*, which is connected to a camera. The thread receives video frames from the camera for feeding them into a memory buffer located inside the controller thread. At grab time, each frame is wrapped inside a class object, which holds the necessary metadata: a time-stamp and flags indicating whether each of the processing stages (face detection, attributes recognition, and similarity search) has been applied on the frame.

B. Face Detection

The first processing step in the pipeline is to find all faces in the input frame. The detection is executed in a dedicated thread, which operates asynchronously, continuously requesting new non-processed frames from the controller thread. The detection algorithm is discussed in detail in Section III-A. Finally, the coordinates of the bounding boxes of all found faces are sent to the controller thread. The controller thread stores the locations and matches each new face with all face objects from the previous frames using straightforward centroid tracking. Tracking allows the system to temporally average the estimates (age, gender, and smile) for each face over a number of recent frames to improve the resulting accuracy.

C. Facial Attributes Recognition

The recognition thread is responsible for assessing the age, gender, facial expression, and facial similarity of each face crop found from the image. Like the detection thread, the recognition thread also operates in an asynchronous mode, requesting new non-processed (but face-detected) frames from the controller thread. When a new frame is received, the thread first aligns the face with a face template. After alignment, we pass each aligned face to separate networks: age, gender, and expression recognizer or a multitask and a similarity search.

Typically, the networks executed on the face crops are slower than the detection network. On the other hand, the amount of time grows linearly with the number of detected faces in the scene. Therefore, in order for the system to appear fast and responsive, these tasks should run in the background and only refresh when each task finishes. More specifically, we refresh the camera view and face detection in real-time but update the recognition results at less than the real-time rate. Moreover, the recognition thread prioritizes the facial expression task over others because age, gender, and facial similarity can be assumed to be constant, while users expect a quick response to their expressions.

The system is implemented using the TensorFlow and OpenCV libraries. The proposed facial analysis architecture can run on various hardware configurations, exploiting either CPU or GPU hardware. As shown in Section V, common desktop hardware reaches real-time speed both on CPU and GPU. However, if the camera resolution, detector type, or input resolution are changed, then a GPU can be used instead.

III. System Components

A. Face Detection

Face detection is the first step for facial recognition systems, where the location of the face is extracted from the given image. We design a neural network based face detector trained using benchmark face datasets. The detectors are not initialized from scratch but fine-tuned from existing pre-trained weights. We experimented with several models from two neural network based detection model categories: single-stage and two-stage detection networks. The single-stage Single Shot Detector(SSD) [12] requires only a single pass through the network with the image as the input and target bounding boxes with respective confidences as the outputs. The two-stage Regions Convolutional Neural Network (RCNN) [13] operates in two stages: a region proposal network proposes candidate object locations, followed by a classifier that classifies the proposals to target categories.

These two structures represent two widely used architectures, where the two-stage RCNN is traditionally perceived as more accurate, especially with small targets. On the other hand, the SSD type networks are simpler, reach faster execution time, and still achieve a reasonable accuracy when the targets are not exceptionally small. However, recent improvements [14], [15] in single-stage detectors have brought single-stage and two-stage architectures closer to each other, both in terms of accuracy and execution speed.

SSD model together with feature extractor networks such as MobileNetV1 [9] and MobileNetV2 [10] are popular for faster and light-weight object detection. MobileNetV1 introduced a parameter α called width multiplier to build a smaller and computationally efficient model. This width multiplier has the effect of reducing computational cost and the number of parameters quadratically by roughly α^2 times. MobileNetV2 introduced a mobile-friendly variant SSDLite that replaces regular convolutions with separable convolutions in the SSD prediction layers, reducing both parameter count and computational cost.

B. Alignment

We align the faces in two stages. The first stage locates a set of facial keypoints from the face crop: eyes, nose, and the corners of the mouth. In the second stage, we find an affine mapping between these five keypoint locations and the corresponding template of five keypoints. This improves accuracy since the recognizers always see the eyes, mouth, and other facial elements in fixed locations, and require less effort in understanding the context where facial features are located. This also enables the use of smaller networks, which compensate for the added computation due to the alignment procedure.

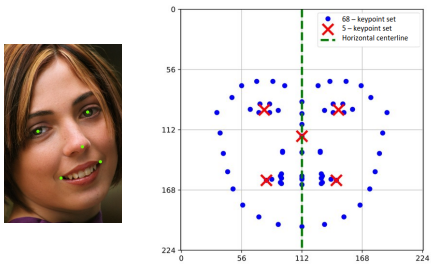

Fig. 3. An example of five-point facial keypoint on a cropped face region (left) and keypoint template (right). Symmetric keypoints are in blue dots, and the 5 referenced true keypoints are highlighted with orange color.

Keypoint Detection—The intention of aligning the faces to fixed coordinates is that this should improve the prediction accuracy. To this aim, we first find the keypoints for each face detected by the detector. We use five facial keypoints for normalizing the face location: eyes, nose, and the corners of the mouth, as illustrates in Fig. 3. Among the accurate and lightweight keypoint detection techniques, we consider regression forests of Kazemi *et al.* [16] and a convolutional neural network, where both receive the face crop as input and output the predicted x-y-coordinates of the five keypoints. We design the convolutional network according to the keypoint location branch of the O-Net [17]; consisting of four convolutional layers and two fully connected layers. The facial keypoint detector is trained from scratch on AFLW dataset [18].

Affine Mapping—The detected keypoints are aligned to a set of template keypoints. The template is obtained from the keypoints of a randomly selected sample face from the dataset. However, we normalize the template such that the keypoints are horizontally symmetric with respect to the centerline of the face. This is done in order to allow training set augmentation by adding horizontal flips of each training face. More specifically, we manually marked symmetric pairs of keypoints and averaged their vertical coordinates and distances from the horizontal center location as illustrated in Fig. 3. Finally, the resulting set of coordinates was scaled to fit the network input size of 224×224 pixels, leaving 10% margin at the bottom edge and 20% margin at the other edges.

Face Alignment—Instead of the simple approach of using the full affine transformation with least squares fit, we choose to use a more restricted *similarity transformation* allowing only rotation, scale, and translation, but not shearing. This is due to the possible distortion of the facial shape and the subsequent degradation of the estimation performances.

The similarity transformation $\boldsymbol{H}$ that maps 2D coordinate points $\mathbf{u} \in \mathbb{R}^2 \mapsto \mathbf{v} \in \mathbb{R}^2$ with translation $\mathbf{t} = (t_x, t_y)^T$, scaling $s \in \mathbb{R}_+$ and rotation matrix $\mathbf{R}$ with rotation angle $\theta \in [-\pi, \pi]$ is given by

$$\mathbf{v} = \boldsymbol{H}\mathbf{x} = \begin{bmatrix} s\boldsymbol{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{u} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \mathbf{u} \quad (1)$$

Estimation of the transformation parameters— $\mathbf{R}$, $\mathbf{t}$ and s— can be obtained from the vector cross product of point correspondences in homogeneous coordinates [19]. Given x-y-coordinates of detected keypoints $\mathbf{u}_i = (x_i, y_i, 1)^T$ and corresponding template locations $\mathbf{v}_i = (x'_i, y'_i, 1)^T$ for $i = 1, 2, \ldots, P$ (with at least $P = 2$ correspondences), the least squares solution for $\boldsymbol{H}$ can be obtained from the equation

$$\mathbf{v}_i \times \boldsymbol{H}\mathbf{u}_i = 0. \tag{2}$$

Substituting Eq. (1) into Eq. (2), the system is further simplified to [20]

$$\begin{bmatrix} -y_i & -x_i & 0 & 1 \\ x_i & -y_i & 1 & 0 \end{bmatrix} \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} -y'_i \\ x'_i \end{bmatrix}, \tag{3}$$

which can be solved by the singular value decomposition [19]. Finally, we construct the similarity matrix $\boldsymbol{H}$ by inserting the four solved scalar unknowns into it.

C. Age Estimation

Age estimation is commonly treated as a regression problem. However, in our system, we treated this as a classification task as our system predicts ages among 101 classes. The network is initialized using ImageNet [21] pre-trained weights and fine-tuned in two stages: first with the large but noisy 500K IMDB-WIKI dataset [22] and then using the small but accurate CVPR2016 LAP challenge dataset [23].

D. Gender and Expression Recognition

The gender recognition network is trained from scratch in two stages: first with the 500K IMDB-WIKI dataset and then fine-tuned with the CVPR2016 LAP challenge dataset, same as in the age recognition step.

In our system, we focused only on smile recognition, a binary classification task, detecting smile and non-smile. The smile recognition network is initialized with ImageNet pre-trained weights and fine-tuned with Genki4k dataset [24].

E. Facial Similarity Search

In addition to the age, gender, and facial expression, the fourth analysis task integrated into the system is the facial similarity search. It is currently implemented for demonstrating *celebrity search*, *i.e.*, the program holds a database of celebrity faces and displays the one whose face has the most similar appearance to the person in front of the demo system. Alternatively, this functionality could be altered to keep track of users using a dynamic database (persons are added to the database every time they are seen) instead of a fixed database (a static collection of celebrities).

The facial similarity search is implemented in two stages: (1) the first stage computes a feature vector from the facial crop using a convolutional network, and (2) the second stage performs the nearest neighbor search among the database of precomputed feature vectors from celebrity faces. We use the FAISS implementation from Facebook [25], since it is widely adopted, provides an interface in Python, and satisfies the real-time speed requirement even with large databases.

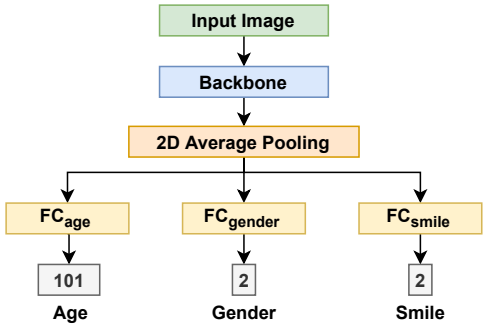

Fig. 4. The architecture of our multitask classification network. The network is able to classify age, gender, and smile attributes for a given image.

We adopt a person re-identification framework [26] to find the most similar face from a collection of celebrity faces. The backbone models are initialized with ImageNet pre-trained weights, and a global average pooling layer is appended to squeeze the spatial dimensions. Several data augmentation policies are applied to make the model more robust, including random flipping, cropping, and random erasing [27]. At the early stage of the training, the learning rate starts from a relatively low value and increases gradually. Additionally, the learning rate gets reduced to one-tenth once the performance on the validation split plateaus.

F. Multitask Network

We experimented with a single multitask network architecture shown in Fig. 4 for age, gender, and smile predictions. The multitask network utilizes a transfer learning approach; backbone network (ImageNet trained) weight is used to fine-tune on the age, gender, and smile training data. The backbone network can be any neural network for the classification task. The last layer of the network is removed, the global average pooling layer is added, and the output from the pooling layer is split into three branches. The fully connected layers, each of dimension 512 is added and SoftMax of different dimensions is applied for task-specific output branches.

Our multitask network inference time is almost identical to individual classification networks; hence this network is about three times faster than the individual networks for age, gender, and smiles recognition tasks, as reported in Table V.

IV. Experiments

A. Datasets

AFLW— The Annotated Facial Landmarks in the Wild (AFLW) [18] is a large-scale dataset of (25K) face images collected from Flickr. It has 21 landmarks annotations per face. We use this dataset to train 5 – keypoint set detection.

CelebA— The CelebFaces Attributes (CelebA) [28] is a large-scale face attributes dataset containing more than 200K celebrity images from 10,177 identities. It has 5 landmark locations and 40 binary attribute labels per facial image.

TABLE I
COMPARISON OF DIFFERENT DETECTION MODELS FOR FACE DETECTION WITH DIFFERENT INPUT SIZES.

Resolution	AP 0.5:0.95	AP @0.5	FPS TF-CPU	FPS TF-GPU	FPS OpenCV
Faster RCNN ResNet101					
300x300	**0.747**	**0.945**	1.84	7.62	1.09
240x180	0.707	0.914	1.93	8.18	1.20
200x200	0.693	0.907	**1.98**	**8.30**	**1.21**
SSD MobileNetV1 $\alpha = 1$					
300x300	**0.744**	**0.945**	32.52	87.29	39.36
240x180	0.684	0.868	**51.60**	105.46	70.09
200x200	0.683	0.839	49.96	**107.54**	**71.42**
SSD MobileNetV1 $\alpha = 0.25$					
300x300	**0.695**	**0.909**	60.50	148.77	140.95
240x180	0.647	0.895	**81.29**	156.62	239.59
200x200	0.650	0.887	78.21	**158.20**	**249.72**
SSDLITE MobileNetV2 $\alpha = 1$					
300x300	**0.764**	**0.952**	28.46	79.94	36.47
240x180	0.728	0.936	**41.18**	106.98	63.29
200x200	0.730	0.934	40.50	**109.47**	**64.79**
SSDLITE MobileNetV2 $\alpha = 0.25$					
300x300	**0.733**	**0.936**	43.09	118.29	70.58
240x180	0.704	0.925	60.43	129.25	125.55
200x200	0.679	0.913	**64.01**	**131.77**	**131.22**

ChaLearn LAP— We use the CVPR2016 competition variant, consisting of 7,591 facial images with human-annotated apparent ages and standard deviations taken in non–controlled environments with diverse backgrounds.

Genki-4k— The MPLab Genki-4k [24] contains 4,000 images with two class expressions (smile or non-smile) labeled by human and head-pose labels of the faces determined by automatic face detector.

B. Evaluation Metrics

AP—The Average Precision (AP) metric computes the average precision overall detection thresholds. The sensitivity of the detector can be adjusted using a detection threshold set by default at 0.5. As the sensitivity of detection may be adjusted at the inference process, we also average the class-wise AP's over all classes to produce the mean AP (mAP).

MAE—The Mean Absolute Error (MAE) metric computes the average error overall prediction. We used MAE to measure the error (in years) at the age prediction stage.

Accuracy—Accuracy is the fraction of correctly classified instances among the total number of instances.

CMC rank-k accuracy—Given a query sample, the accuracy is set to 1 if the top-k gallery samples contain samples that have the same identity as the query sample, and 0 otherwise. The CMC rank-k accuracy is obtained by averaging the results of all query samples.

V. RESULTS AND DISCUSSION

A. Face Detection

For face detection, we use faster RCNN, with ResNet101 backbone and variants of SSD, with MobileNet backbones with three different input sizes. The network inference speed is tested on Tensorflow CPU and GPU, and OpenCV environments, illustrated in Table I. The faster RCNN ResNet101 network in all experiments gives slightly higher AP than SSD models. However, the computation complexity of RCNN models is high, i.e., lower FPS compared to one-stage networks. Experiments show that with $\alpha = 0.25$, detection performance is about 3% less accurate while increasing inference speed about 1.5 times.

For all experimented networks, optimal detection performance is obtained by larger input size (*i.e.*, 300×300), while best FPS is obtained with smaller input size (*i.e.*, 200×200). With a smaller square input size, using OpenCV at inference always guarantees the best inference speed. If detection accuracy is not the top priority, using a small value of the α, smaller input size, and lighter model is suitable for faster and memory-efficient detection.

TABLE II
KEYPOINT DETECTION PERFORMANCE.

	Error rate(%)	FPS CPU
Dlib	2.89	17.49
CNN	**1.04**	**18.25**

TABLE III
PERFORMANCE IN FACIAL SIMILARITY USING THREE BACKBONES.

Network	mAP	rank-1	rank-5
MobileNet	0.782	0.940	0.970
VGG16	0.813	0.952	0.973
ResNet50	**0.822**	**0.953**	**0.973**

We measured the performance of two keypoint detection methods on the AFLW dataset. During the training, keypoint detection models were optimized based on the detected facial area with ground-truth keypoint labels.

Experiments on O-Net [17] based CNN alignment gives better alignment accuracy and inference speed as shown in Table II.

B. Facial Similarity

Our facial similarity system aims at finding the most similar face, and the rank-1 accuracy is a preferable evaluation metric. Table III shows the mAP, rank-1 accuracy, and rank-5 accuracy of facial similarity models trained with categorical cross-entropy loss on the aligned images from the CelebA dataset. The rank-1 accuracy of MobileNet reaches 94.0% which is slightly inferior to VGG16 and ResNet50, while using MobileNet is computationally lightweight.

C. Age, Gender and Expression Recognition

Table IV shows the accuracies of the different tasks included in our system. The speed test of each task on two different environments indicates that in the same environment, there is no significant difference between the network in terms of inference speed. However, the multitask network appears better considering the total inference time for three tasks.

The experimental results on our multitask network for age estimation, gender, and smile recognition with different backbone networks are reported in Table V. The best performing multitask network gives 5.35 years age MAE which is slightly higher than the best performing individual age network. Also, gender and smile accuracies obtained from the best performing multitask network are slightly less accurate than the individual networks. EfficientNet [11] networks give better age MAE and

TABLE IV
ACCURACIES AND INFERENCE SPEED AT DIFFERENT STAGES IN OUR SYSTEM. THE DEPTH MULTIPLIER $\alpha = 1.0$ IS USED IN ALL MOBILENETV1 NETWORKS.

Stage Network	Accuracy	FPS CPU	FPS 1050 TI	FPS 1080 TI
Age MobileNetV1	4.9 MAE	31.61	148.90	147.44
Gender MobileNetV1	88.3%	31.48	150.45	149.75
Smile MobileNetV1	87.2%	31.46	148.84	148.78
Multitask MobileNetV1	5.67 MAE 84.2% Gender 83.6% Smile	29.80	147.06	147.20
Multitask EfficientNetB0	5.35 MAE 87.5% Gender 86.0% Smile	25.61	143.35	144.24

TABLE V
PERFORMANCE COMPARISON OF THE MULTITASKING NETWORK WITH DIFFERENT BACKBONE NETWORKS.

Network	Age MAE	Gender Acc(%)	Smile Acc(%)	FPS CPU
VGG16	7.20	84.0	84.1	27.75
ResNet50	6.42	82.1	81.2	27.06
ResNet18	6.02	82.4	82.8	29.63
MobileNetV1	5.67	84.2	83.6	**29.80**
EfficientNet B0	5.35	87.5	86.0	25.61
EfficientNet B1	5.07	87.8	86.8	22.72
EfficientNet B7	**4.37**	**89.5**	**87.3**	14.63

recognition accuracies, but they are computationally expensive. As our computational budget is limited, and we cannot use a combination of many networks.

We can set up a system with a combination of a lighter detection model and a faster multitask network if the network accuracies are not the top priority. This way, real-time inference speed can be achieved on the CPU while slightly compromising each task's accuracy.

VI. CONCLUSION

We present a system-level design of a human facial analysis system with a multi-threaded architecture to reach real-time operation on resource-limited devices. We describe individual components of our system, integrating several standard machine learning components with an extensive set of experiments on each task. Users can switch specific task networks from the list of available options on the fly. The demo system has been presented several times in public locations. It has shown its value in illustrating the potential of modern machine learning in an easy-to-approach use case working on many levels. Moreover, this system can be used in the surveillance system by adding an alarm function that triggers once the detected face is matched with the suspect's faces on the query dataset. Additionally, the system can be used as a reference and baseline for related applications.

REFERENCES

[1] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *CVPR*, 2014.

[2] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *ICCV*, 2009.

[3] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation in computer vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, 2009.

[4] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *CVPR*, 2011.

[5] C. Shan, S. Gong, and P. W. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," *Image and Vision Computing*, vol. 27, no. 6, pp. 803 – 816, 2009.

[6] P. Liu, S. Han, Z. Meng, and Y. Tong, "Facial expression recognition via a boosted deep belief network," in *CVPR*, 2014.

[7] Y. Dong, Y. Liu, and S. Lian, "Automatic age estimation based on deep learning algorithm," *Neurocomputing*, vol. 187, pp. 4–10, 2016.

[8] J. Yrjänäinen, X. Ni, B. Adhikari, and H. Huttunen, "Privacy-aware edge computing system for people tracking," in *International Conference on Image Processing (ICIP)*, 2020, pp. 2096–2100.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[10] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018.

[11] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *(ICML)*, ser. Proceedings of Machine Learning Research. PMLR, 2019.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *ECCV*, 2016.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[14] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017.

[15] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *ICCV*, 2017.

[16] V. Kazemi and S. Josephine, "One millisecond face alignment with an ensemble of regression trees," in *CVPR*, 2014.

[17] 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, pp. 1499–1503, 2016.

[18] P. M. R. Martin Koestinger, Paul Wohlhart and H. Bischof, "Annotated Facial Landmarks in the Wild: A Large-scale, Real-world Database for Facial Landmark Localization," in *Proc. First IEEE International Workshop on Benchmarking Facial Image Analysis Technologies*, 2011.

[19] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[20] J.-K. Kamarainen and P. Paalanen, "Experimental study on fast 2D homography estimation from a few point correspondences," Lappeenranta University of Technology, Tech. Rep. 111, 2009.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[22] R. Rothe, R. Timofte, and L. V. Gool, "Deep expectation of real and apparent age from a single image without facial landmarks," *International Journal of Computer Vision (IJCV)*, 2016.

[23] S. Escalera, M. T. Torres, B. Martínez, X. Baró, H. J. Escalante, I. Guyon, G. Tzimiropoulos, C. Corneanu, M. Oliu, M. A. Bagheri, and M. Valstar, "Chalearn looking at people and faces of the world: Face analysis workshop and challenge 2016," in *CVPRW*, 2016.

[24] S. D. MPLab, Uiversity of California, "The MPLab GENKI Database, GENKI-4K Subset."

[25] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *IEEE Transactions on Big Data*, 2019.

[26] H. Luo, Y. Gu, X. Liao, S. Lai, and W. Jiang, "Bag of tricks and a strong baseline for deep person re-identification," in *CVPRW*, 2019.

[27] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[28] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *ICCV*, 2015.