# Elastic Neural Networks: A Scalable Framework for Embedded Computer Vision

Yue Bai
*Tampere Univ. of Tech.*
Tampere, Finland

Shuvra S. Bhattacharyya
*University of Maryland, USA, and Tampere Univ. of Tech., Finland*

Antti P. Happonen
*Tampere Univ. of Tech.*
Tampere, Finland

Heikki Huttunen
*Tampere Univ. of Tech.*
Tampere, Finland

***Abstract*—We propose a new framework for image classification with deep neural networks. The framework introduces intermediate outputs to the computational graph of a network. This enables flexible control of the computational load and balances the tradeoff between accuracy and execution time.**

**Moreover, we present an interesting finding that the intermediate outputs can act as a regularizer at training time, improving the prediction accuracy. In the experimental section we demonstrate the performance of our proposed framework with various commonly used pretrained deep networks in the use case of apparent age estimation.**

***Index Terms*—Deep learning, machine learning, regularization, embedded implementations, age estimation**

## I. Introduction

Deep learning has rapidly become the state of the art in modern machine learning, and has surpassed the human level in several classical problems. Majority of research concentrates on the accuracy of the model, but also the computational load has been studied. This line of research attempts to reduce the execution time of the forward pass of the network. Well-known techniques for constructing lightweight yet accurate networks include decomposing the convolutions either in horizontal and vertical dimensions [1], or in spatial and channel dimensions [2], or using a fully convolutional architecture [3].

In non-static environments, the computational requirements may have a large variation. For example, in the use case of this paper—real-time age estimation—the number of faces seen by the camera directly influences the amount of computation. Namely, each detected face is sent to the GPU for age assessment. Thus, the workload for 10 faces in front of the camera is 10-fold compared to just a single face. Additionally, the same network should be deployable over a number of architectures: For example, in 2015 there were already over 24,000 different hardware architectures running the Android operating system. Thus, there is a demand for an *elastic network structure*, able to adjust the computational complexity on the fly. The network should be monolithic (as opposed to just training several networks with different complexities), since the storage and computational overhead of deploying a large number of separate networks would be prohibitive in many applications.

This work was partially funded by the Academy of Finland project 309903 CoefNet, and by TEKES — the Finnish Technology Agency for Innovation (FiDiPro project StreamPro 1846/31/2014). Authors also thank CSC–IT Center for Science for computational resources.

Recently, Huang *et al.* [4] proposed a network architecture addressing these problems. Their *Multi-Scale Dense Network* (MSDNet) adds early-exits to the computational graph, and stores a representation of the input at several resolutions throughout the network. This way, the computational time can be adjusted even to the extent of *anytime prediction*, where the network almost always has some prediction at hand.

In this work, we take a simpler approach: Instead of proposing a new network topology, we study the effect of adding early-exits to *any network*. Namely, there exist a number of successful network topologies pretrained with large image databases and the progress of deep learning is so rapid, that new network topologies appear daily. We propose a general framework that systematically "elastifies" an arbitrary network to provide novel trade-offs between execution time and accuracy, while leveraging key properties of the original network.

## II. Elastic Deep Neural Networks

In deep neural networks, the time to complete the forward (prediction) pass is composed of the execution times of individual layers. As an example, Figure 2 shows the cumulative floating-point operations (FLOPs) per image frame (i.e., for each iteration of the neural network) for different state-of-the-art network structures. Conventionally, the network outputs the class probabilities of the input image at the final classifier layer, thus summing up the number of FLOPs spent at the early layers. The final classifier layer is also the most accurate one to categorize the input. However, the feature maps of the earlier layers also possess some predictive power, leading to the question of whether the early layer features are useful for prediction, as well. Thus, we introduce a classifier structure with early-exits, which base their prediction on the features extracted at intermediate layers. An illustration of the proposed structure is shown in Figure 1. The figure illustrates the elastic structure in an apparent age estimation context, where the input RGB image has dimensions $224 \times 224 \times 3$ and the prediction target is a vector of probabilities for ages $y \in \{0, 1, 2, \ldots, 100\}$. In particular, the figure shows the early exits, which are labeled Prediction 1, Prediction 2, ..., Prediction 7.

For each early exit (and the final one), we set exactly the same training target and loss function. More specifically, our loss function for a training sample $(\mathbf{x}, \mathbf{y})$, with input $\mathbf{x} \in$

ISBN 978-90-827970-1-5 © EURASIP 2018

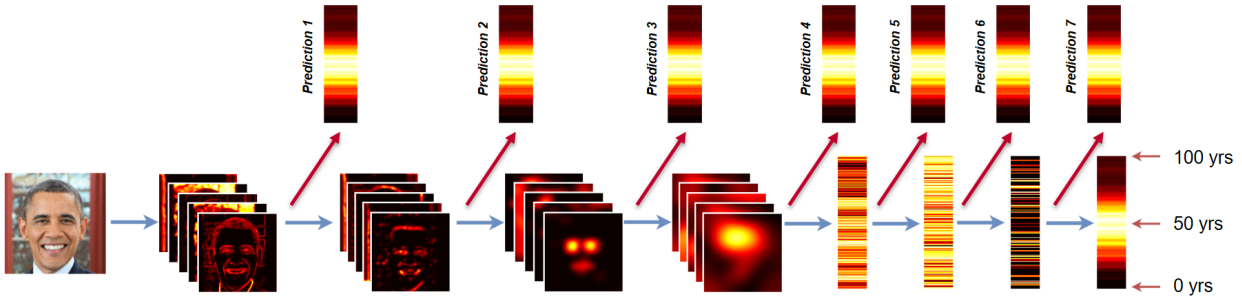
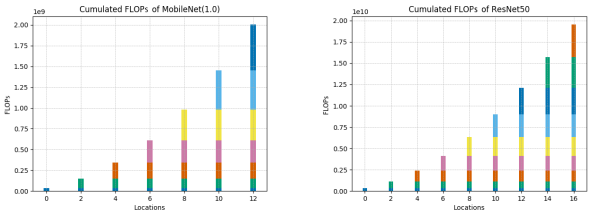



Fig. 1: The proposed neural network structure with early exits.




(a) (b)

Fig. 2: Cumulative FLOPs on MobileNet ($\alpha = 1$) and ResNet50.

$\mathbb{R}^{w\times h\times d}$ and one-hot-encoded class label $\mathbf{y} \in \mathbb{R}^C$ (with $C$ classes), is defined as:

$$L(\mathbf{w};\mathbf{x},\mathbf{y}) = \sum_{p=1}^{P} \alpha_p \ell(\mathbf{y}, \hat{\mathbf{y}}_p(\mathbf{x};\mathbf{w})), \qquad (1)$$

where $\hat{\mathbf{y}}_p(\mathbf{x};\mathbf{w})$ denotes the $p$'th output in the pipeline ($p = 1, 2, \ldots, P$) for network with parameters $\mathbf{w}$ and input $\mathbf{x}$; $\alpha_p > 0$ is the weight for each output; and $\ell(\cdot,\cdot)$ denotes a loss function (*e.g.,* categorical cross-entropy).

In practice, the number of layers may be high (even hundreds of layers), and the number of early exits may grow unnecessarily high and slow down the training process if an early exit is inserted after every layer. However, most modern networks are constructed of repeated blocks, where each block contains multiple layers. By placing early exits only at the block outputs, we can significantly reduce the burden on the training process. Moreover, the very early blocks are not useful for prediction, so we assign zero weights $\alpha_p = 0$ to those early exit losses.

## III. Apparent Age Estimation

We investigate the performance of the proposed elastic networks methodology in real-time apparent age estimation. Figure 3 shows an example of the application which is related to our real time age estimation demo [5], where people may freely enter in front of a screen and a camera, and the system then detects all faces and assesses the age, gender and facial expression of each based on the cropped bounding box of each face. When multiple people appear in front of the screen, the system has to evaluate the ages for each face separately, and

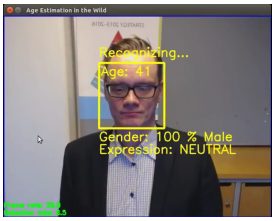



Fig. 3: Real time age, gender and facial expression recognizer.

the computational load increases accordingly. Thus, the need for flexible control of the computational load is evident.

Let us now describe the detailed training protocol: datasets, preprocessing, and the network structures used.

### A. Datasets

Following the procedure of [6], we use altogether three datasets for training. First, we initialize our network with Imagenet [7] pretrained weights, then we pretrain the network for faces using the IMDB-WIKI [6] facial database (large but noisy), and finally train the network using a smaller human annotated ChaLearn LAP dataset from CVPR2016 competition (small but clean).

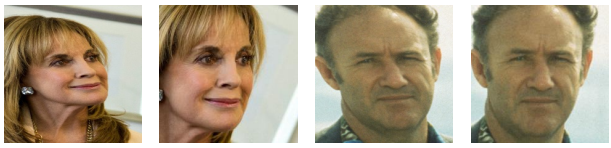
***IMDB-WIKI***—The IMDB-WIKI dataset is the largest publicly available dataset of facial images with gender and age labels for training. The facial images in this dataset are crawled from the IMDB and Wikipedia websites. There are 461,871 facial images from the IMDB website and 62,359 from the Wikipedia website. At each epoch, we feed the network with 32,768 randomly chosen images with left-right-flip augmentation and iterate for 100 epochs.

***CVPR 2016 ChaLearn Looking at People Competition on apparent age estimation dataset***—In the apparent age estimation competition organized at IEEE CVPR 2016 [8], a state-of-the-art dataset was released with altogether 7,591 facial images (4,113 in the training dataset, 1,500 in the validation dataset and 1,978 in the testing dataset) with human-annotated apparent ages and standard deviations. Compared to other facial age datasets, the images in this dataset are taken in non-controlled environments and have diverse backgrounds.

### B. Image Preprocessing

***Face detection***—During the real-time prediction, we use the Viola-Jones [9] face detector. Despite its poor accuracy

ISBN 978-90-827970-1-5 © EURASIP 2018



(a) (b) (c) (d)

Fig. 4: The result of alignment with full affine transformation is shown in (a) and (c), while (b) and (d) depicts the similarity transformation without shearing.

compared to modern CNN detectors, it is still among the fastest algorithms when not using the GPU (which we wish to reserve for the age estimation). The deficiencies in accuracy are compensated by the streaming nature of the data source; not every frame has to be detected.

At training time, however, we want to exploit every training image—even those at poses difficult for the Viola-Jones detector. Therefore, We use the open source 'Head Hunter' [10] face detector to detect the bounding boxes. If there are multiple faces are detected in one image, we will choose the first one or the one with the highest score. Note that the discrepancy of using a different detector at training and testing time is compensated by the subsequent alignment step, where we align the face to a template set of facial landmarks.

***Landmark Detection***—Alignment of the face to fixed coordinates will greatly improve the prediction accuracy. To this aim, we will first find the landmarks for each face found by the detector. In our implementation, we use the regression forest method of Kazemi *et al.* [11], due to its fast speed. In our landmark set, we use the *dlib*[1] implementation with 68 landmarks, of which we use 47 central keypoints, discarding the boundary keypoints often occluded due to pose. We match the found facial keypoints with a set of target keypoints by using a similarity transformation matrix.

***Target Landmarks***—The landmark template is obtained from the landmarks of a sample face from the dataset. However, we normalize the template such that the landmarks 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 landmarks, and averaged their vertical coordinates and distances from the horizontal center location. Finally, the resulting set of coordinates was scaled to fit the network input size of $224 \times 224$ pixels leaving 10% margin at the bottom edge and 20% margin at the other edges. The resulting landmark template is illustrated in Figure 5.

***Face Alignment***—The simple approach of using the full affine transformation with least squares fit will distort the image and degrade the estimation performance. This is due to the shearing component as shown in Figure 4a and Figure 4c. This way the shape of faces in original images can be affected badly, which we wish to avoid. Instead, we use the similarity transformation allowing only rotation, scale and translation. The definition of the similarity transformation mapping 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 angle $\theta$ is:

$$\mathbf{v} = \boldsymbol{H}_S \mathbf{x} = \begin{bmatrix} s\boldsymbol{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{u} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \mathbf{u} \quad (2)$$

According to Hartley and Zisserman [12], the solution can be derived from the vector cross product of point correspondences in homogeneous coordinates, $\mathbf{u}_i = (x_i, y_i, 1)^T$ and $\mathbf{v}_i = (x_i', y_i', 1)^T$, as

$$\mathbf{v}_i \times \boldsymbol{H}_S \mathbf{u}_i = 0. \quad (3)$$

Substituting Eq. (2) into Eq. (3), the system simplifies to [13]

$$\begin{bmatrix} -y_i & -x_i & 0 & 1 \\ x_i & -y_i & 1 & 0 \end{bmatrix} \begin{bmatrix} s\cos\theta \\ s\sin\theta \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} -y_i' \\ x_i' \end{bmatrix}, \quad (4)$$

which can be solved by the singular value decomposition [12]. Finally, we construct the similarity matrix $\boldsymbol{H}_S$ by inserting the four variables into it. Examples of aligned pictures are shown as in Figure 4b and Figure 4d.

### C. Pre-Trained Networks

We use three commonly used network structures, each pretrained with the 1000-class ImageNet dataset. For age estimation, we only take the convolutional pipeline of each and append a 101-class dense layer to the top (for 101 ages: $0, 1, \ldots, 100$ years). In this section, we present three elastic network structures that we developed for apparent-age estimation, and we specify where the early exits are located in these networks.

***ResNet50 Based Elastic Neural Network***—ResNet50 [14] has 50 convolution layers, composed mainly of 16 three-layer residual blocks. We insert early-exit classifiers after each residual block producing a total of 17 intermediate classifiers (plus the final classification layer) and set the weights nonzero starting at the midpoint of the pipeline: $\alpha_p = 0$ for $p < 9$ and $\alpha_p = 1$ for $p \geq 9$.

***MobileNet Based Elastic Neural Network***—MobileNets [2] have altogether 28 convolutional layers, consisting of an input layer, 13 two-layer blocks (one depthwise and one pointwise convolution) and the output layers. We connect early-exit classifiers to the outputs of all the 13 blocks and the final classification layer and set the weights nonzero starting at the midpoint of the pipeline: $\alpha_p = 0$ for $p < 7$ and $\alpha_p = 1$ for $p \geq 7$.

***Inception-V3 Based Elastic Neural Network***—Inception-V3 network [15] has altogether 94 convolutional layers (some in parallel) include 11 blocks. We insert early-exit classifiers after each blocks producing a total of 12 intermediate classifiers (plus the final classification layer) and set the weights nonzero starting at the midpoint of the pipeline: $\alpha_p = 0$ for $p < 6$ and $\alpha_p = 1$ for $p \geq 6$.

[1] http://dlib.net/

ISBN 978-90-827970-1-5 © EURASIP 2018

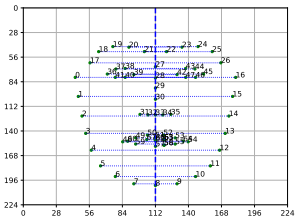


Fig. 5: Landmark targets for the alignment. Symmetric landmarks pairs are highlighted by dashed lines.

## IV. Experimental Results

We implement our proposed models using Keras based on Tensor flow backend [16]. All models are trained using stochastic gradient descent (SGD) with mini-batch size 32. We use momentum with a momentum weight of 0.9 .

The initialized weights for every output layer follow the uniform distribution. We give a weight for every output layer equal to one. During the pretraining phase, we first freeze all layers except output layers and train with a learning rate of $1 \times 10^{-3}$ for the first 10 epochs. Then we train all layers of all models for 150 epochs, with an initial learning rate of $1 \times 10^{-2}$, which is divided by a factor of 10 after the decaying loss on the validation set stays on a plateau. The minimum of learning rate is $1 \times 10^{-5}$. We apply the same optimization scheme to the training phase, except that the initial learning rate is decreased to $1 \times 10^{-3}$.

### A. Evaluation Criteria

We use two different criteria to evaluate the accuracy of age estimation.

*MAE*—Mean Absolute Error (MAE) is the average of absolute differences between the predicted and the real age:

$$MAE = \frac{1}{N}\sum_{n=1}^{N}|\hat{y}_n - y_n|, \tag{5}$$

where $\hat{y}_n$ and $y_n$ are the estimated and ground-truth age of the $n$-th testing image, respectively.

***$\epsilon$-error***—The degree of agreement among the human annotators may be used as an indicator of difficulty of each image. In our training dataset, the minimum and maximum standard deviations are 0 and 12.4, while in the validation these are 0 and 14.1. To take the relative difficulty into account, the ChaLearn competition organizers defined an $\epsilon$-metric defined as

$$\epsilon = 1 - \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right),$$

where $y$ is the age predicted by the algorithm, $\mu$ is the mean assessment of human evaluators, and $\sigma^2$ is their variance. We use it as another accuracy criterion in our experiments.

### B. Results

Figure 6 shows the performance of the proposed elastic methodology implemented based on different networks on the ChaLearn LAP 2016 apparent age estimation testing dataset.

TABLE I: Test set accuracies of the ChaLearn LAP 2016 apparent age estimation dataset. The best $\epsilon$-error of each row in bold.

| Model | w/ Elastic Structure | | w/o Elastic Structure | |
|---|---|---|---|---|
| | MAE | $\epsilon$-error | MAE | $\epsilon$-error |
| InceptionV3 | 4.0070 | **0.3279** | 4.3967 | 0.3592 |
| ResNet50 | 4.1016 | **0.3428** | 4.5221 | 0.3682 |
| MobileNet ($\alpha = 1$) | 4.7174 | **0.3834** | 5.1672 | 0.4097 |
| MobileNet ($\alpha = 0.75$) | 4.9952 | **0.4022** | 5.6941 | 0.4311 |
| MobileNet ($\alpha = 0.5$) | 6.0504 | **0.4296** | 6.3596 | 0.4633 |
| MobileNet ($\alpha = 0.25$) | 6.3953 | **0.4518** | 6.3452 | 0.4753 |
| VGG16 | - | - | 5.7225 | 0.4338 |
| CVPR2016 winner [1] | - | - | - | 0.2411 |
| CVPR2016 2nd [1] | - | - | - | 0.3214 |
| CVPR2016 3rd [1] | - | - | - | 0.3361 |

[1] CVPR winners use more complicated preprocessing and/or multiple VGG16 style networks [18]

TABLE II: FLOPs of Elastic Framework for the networks compare to the state of art VGG16 network ($\Delta$ VGG).

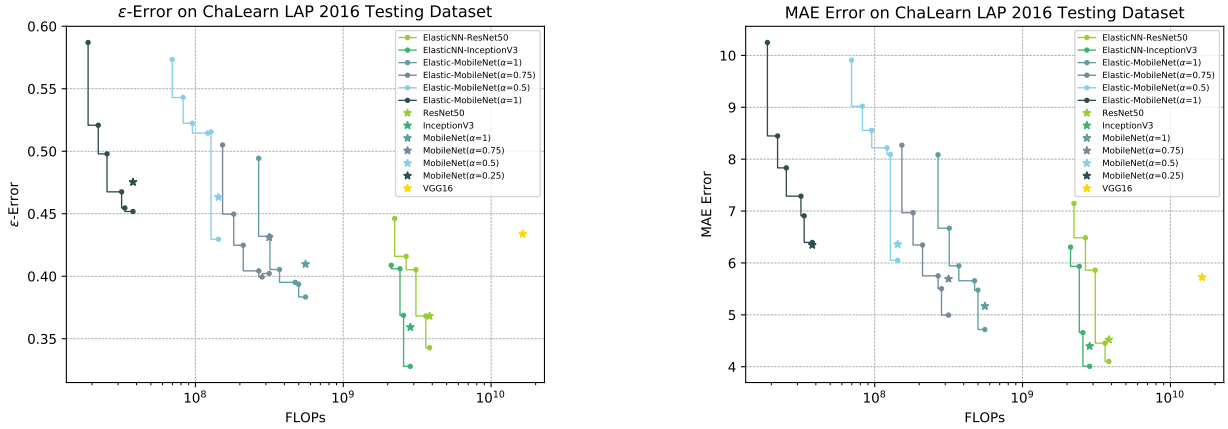
| Model | FLOPs | | $\Delta$ VGG | |
|---|---|---|---|---|
| | first-exit | last-exit | first-exit | last-exit |
| VGG16 (w/o Elastic) | $1.53 \times 10^{10}$ | | 0% | |
| InceptionV3 | $1.87 \times 10^{9}$ | $2.84 \times 10^{9}$ | 87.83% | 81.48% |
| ResNet50 | $2.23 \times 10^{9}$ | $3.83 \times 10^{9}$ | 85.49% | 75.02% |
| MobileNet ($\alpha = 1$) | $2.68 \times 10^{8}$ | $5.56 \times 10^{8}$ | 98.25% | 96.38% |
| MobileNet ($\alpha = 0.75$) | $1.53 \times 10^{8}$ | $3.16 \times 10^{8}$ | 99% | 97.94% |
| MobileNet ($\alpha = 0.5$) | $6.97 \times 10^{7}$ | $1.43 \times 10^{8}$ | 99.55% | 99.07% |
| MobileNet ($\alpha = 0.25$) | $1.88 \times 10^{7}$ | $3.78 \times 10^{7}$ | 99.88% | 99.75% |

In particular, our elastic structures consistently exceed the performance of the corresponding original network structures. Table I displays the result at the situation with and without elastic structure separately. It shows that both MAE and $\epsilon$-error decrease notably for InceptionV3, ResNet50, and MobileNet ($\alpha = 1/0.75/0.5$, the $\alpha$ parameter in MobileNet increases or decreases the number of filters in each layer) on elastic structures. This indicates an interesting result that by introducing intermediate outputs act as regularizers, which benefits the final layer result.

Among elastic neural network architectures we carry out, InceptionV3 based network achieves the best performance with MAE and $\epsilon$-error equal to 4.01 and 0.3279 separately. Comparing to the result of the ChaLearn LAP 2016, our result has reached the third place. Notably, We only use a single network instead of combining the strength of different networks, besides we use a rather fast image preprocessing method which are more suitable for real-time tasks.

Table II compares elastic neural network architectures in terms of their FLOPs requirements. It shows the FLOPs on the first intermediate output and on the last output separately of every elastic neural network architectures. The comparison includes the decreasing rate of FLOPs against the VGG16 [17] neural network ($\Delta$ VGG) on the first and the last exit. The FLOPs on our proposed architectures decrease dramatically compare to VGG16, which is considered as an important benchmark for age estimation.

ISBN 978-90-827970-1-5 © EURASIP 2018




(a) $\epsilon$-error over different networks

(b) MAE over different networks

Fig. 6: Performance on different Elastic Framework based networks. Lower is better.

## V. Discussion and Conclusion

We propose a general framework that systematically "elastifies" an arbitrary network to provide novel trade-offs between execution time and accuracy. This "elastification" is carried out by adding the so-called early-exits or intermediate outputs to any network topology. This framework enables us to design networks with adjustable computational load so that the network structure can elastically adapt to the load on the data stream rate basis. Choosing the intermediate outputs appropriately, we can improve both the computational speed and prediction accuracy compared to full network training, according to our experiments.

In our analysis, we made an interesting finding on the regularization effect of the so-called early exits in different network topologies. The early exits improve network generalization and is strongest with large networks having large expression power. This may open interesting future lines of research in studying if the effect applies in other network structures and applications, as well.

As a use-case, we demonstrate the performance of our proposed framework with various commonly used pretrained deep networks for apparent age estimation. Our results show that the proposed apparent age estimator performs almost equally in comparison with recently reported apparent age estimators, but with significantly less computation.

## References


[1] J. Alvarez and L. Petersson, "Decomposeme: Simplifying convnets for end-to-end learning," *arXiv preprint arXiv:1606.05426*, 2016.

[2] 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.

[3] 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.

[4] G. Huang, D. Chen, T. Li, F. Wu, L. van der Maaten, and K. Q. Weinberger, "Multi-scale dense convolutional networks for efficient prediction," *arXiv preprint arXiv:1703.09844*, 2017.

[5] H. Huttunen, "Deep neural networks: A signal processing perspective," in *Handbook of Signal Processing Systems (Third Edition)*, S. S. Bhattacharyya, E. F. Deprettere, R. Leupers, and J. Takala, Eds. Springer, 2018, to appear.

[6] R. Rothe, R. Timofte, and L. Van Gool, "Dex: Deep expectation of apparent age from a single image," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 10–15.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[8] S. Escalera, M. Torres Torres, B. Martinez, X. Baró, H. Jair Escalante, I. Guyon, G. Tzimiropoulos, C. Corneou, M. Oliu, M. Ali Bagheri *et al.*, "Chalearn looking at people and faces of the world: Face analysis workshop and challenge 2016," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 1–8.

[9] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001.

[10] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *ECCV*, 2014.

[11] V. Kazemi and S. Josephine, "One millisecond face alignment with an ensemble of regression trees," in *27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, United States, 23 June 2014 through 28 June 2014*. IEEE Computer Society, 2014, pp. 1867–1874.

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

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

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[16] F. Chollet, *Deep learning with Python*. Manning Publications, 2018.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[18] S. Escalera, M. T. Torres, B. Martnez, X. Bar, 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 analysisworkshop and challenge 2016," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2016, pp. 706–713.


ISBN 978-90-827970-1-5 © EURASIP 2018