International Journal of Innovative Research in Computer Science & Technology (IJIRCST)
ISSN: 2347-5552, Volume-10, Issue-1, January 2022
https://doi.org/10.55524/ijircst.2022.10.1.4
Article ID IJIR-1148, Pages 15-21
www.ijircst.org

Age and Gender Prediction using Caffe Model and OpenCV

Sharik Shaban[1], Ravinder Pal Singh[2], and Dr. Monika Mehra[3]

[1] M.Tech, Department of Electronics and Communication Engineering, RIMT University, Mandi Gobindgarh, Punjab, India
[2] Associate Professor Department of Research, Innovation & Incubation, RIMT University, Mandi Gobindgarh, Punjab, India
[3] Head of Department, Department of Electronics and Communication Engineering, RIMT University, Mandi Gobindgarh, Punjab, India

Correspondence should be addressed to Sharik Shaban; shabansharik@gmail.com

ABSTRACT-Automatic classification of age and gender has become crucial for a rising number of applications, especially as social platforms and social media have risen. However, there is still substantial lack of performance of present approaches in real-world photos, especially when compared to the enormous performance jumps reported lately in the associated facial recognition job. In this research we show that a considerable gain in performance may be achieved by the application of convolution neural networks (CNN). This work is primarily designed to construct an algorithm that accurately guesses a person's age and gender.Haar cascade is one of the most often utilised approaches. In this research we provide a model that can help Haar Cascade to determine a person's gender. The model trained the classifier as positive and negative pictures using diverse photos of men and women. Various face characteristics are removed. With the help of Haar Cascade, the classifier determines if the picture input is men or women. Even with insufficient data, it functions effectively. A deep education framework created with Caffe is used to do the age or sex approximation task. Our model is able to detect multiple faces in single image and predict age and gender of all faces present in the image.

KEYWORDS- Haar Cascade, Caffe Model, OpenCV, Convolutional Neural Network.

I. INTRODUCTION

Usually interchangeable, the word "facial detection" and "facial recognition" has various significance. Face and face detection are both complex computer vision tasks; yet, face detection is normally the first step in several facial applications that identify the existence, location and size of human faces in digital images, whereas face detection consists mainly of two phases.The process of recognising and locating the face is also viewed as a classifying task which should be properly classified by facial detection techniques (a small False Positive Rate) when an input is presented as a digital picture of various sources that include video, scanner, web and camera, whether a human face is present or missing. Facial analysis in the recent past has become well recognised in the computer-view community[1–4]. The human face includes identity, age, gender, emotions, and people's race characteristics [5, 6].
In my work, I haveused OpenCV and Haar Cascade algorithm with pretrained Caffe Model for age and gender detection.

A. Open CV DNN

It is an OpenCV module. A pre-trained Tensorflow model can be used. This is, however, a deep neural network that may be utilised with a pre-trained model to infer. Support for several frames such as Cafe, Tensorflow, Darknet and PyTorch is provided by OpenCV DNN. Various applications like face detection and object detection may be created using this module.

B. Caffe Model

Caffe is a profound learning context that takes into account expressiveness, speed and modularity. Berkeley AI Research (BAIR) is produced and supported by community contributors. During his PhD at UC Berkeley, Yangqing Jia designed the project. Under the BSD 2-Clause licence, Caffe is released.

C. Caffe Data Storage

In 4-dimensional arrays called blobs coffee saves and transmits data. Blobs give a single memory interface with plenty of pictures, parameters or modifications to parameters (or other information). By synchronising from the CPU host to the GPU devices, the blobs cover the mental and computational overhead of the mixed CPU/GPU process. Usually, data may be loaded from the disc to a CPU blob, the CUDA kernel is called for GPU computing and the blob is shut off to the next layer, disregarding the low level details and retaining its high performance. Host memory and device are assigned to efficient memory utilisation on request.

D. Caffe Layers

A caffe level is the core of a neural network layer: one or more blobs are used as input and one or more blobs are produced as output. Levels have two main duties in the functioning of the whole network: a forward transmission, which takes the inputs and creates the outputs and a return transition, which takes the gradient from the output and calculates the gradient from the parameters and inputs which, in turn, are transmitted back to the previous layers. Including: convolution, pooling, internal product, nonlinearity like linear and logistic corrected, normalisation of local response and losses such as softmax and hinge. These are all kinds of visionary jobs needed for the most advanced. Due to the composition of networks, coding of custom layers needs minimum effort.

E. Face Detection using Viola-Jones Algorithm

Viola Jones algorithm has its name from the work "Rapid Object Detection Using a Boosted Cascade of Simple Features" by two computer vision researchers who presented the approach in 2001, Paul Viola and Michael Jones. Although Viola-Jones is an outdated frame, it is highly strong and is an amazing application for real-time face identification. This algorithm is quite slow to train, yet can recognise faces with amazing speed in real time.

Given a picture, the algorithm looks at numerous smaller sub-regions and tries to discover a face by looking at certain attributes inside each sub-region. It must verify many distinct locations and dimensions since a picture has multiple faces of varied sizes. For detecting faces in the system, Viola and Jones exploited hairlike characteristics.

There are four primary phases in the Viola Jones algorithm, which we address in the following sections:

- Haar-like characteristics
- Create a comprehensive picture
- Training AdaBoost
- Creating cascades for the classifier

1) Selecting Haar-like features

In the 19th century, Alfred Haar, a Hungarian mathematician, presented a sequence of rescaled 'square' functions that formed the basis of a wavelet family. Voila and Jones have adapted and developed the so- called haarlike qualities to the notion of employing hair waves. Digital picture characteristics utilised for object detection are hair-like characteristics. The eyes area is darker than its neighbouring pixels and the nose area is brighter than the eye area. All human faces have some universal features of the human face.

The pixel values of the two areas may be summed up and compared simply to find out whether section is brighter or darker. The pixel value in the darker area is lower than the pixel value in the brighter section. It might be an edge of the eyebrow on one side or it can be shiner in the central part than the surrounding boxes that can be construed as a nose We may do this through the use of hair-like characteristics and interpret the various areas of a face with them.

2) Creating Integral Images

Functions that are important to the study must be selected since this makes the process of detection faster and more accurate. Comprehensive pictures decrease the computer time on an image input by utilising just their four corner values to calculate the total of all pixels in a particular rectangle. Therefore, not all the functional values are computed, instead of certain basic calculations, on the collar pixel input picture values. It is worth noting that the pixel value (x, y) for an integral picture is derived by summing the above and left pixels (x, y). In addition, the final total of the rectangle is computed by removing the total of the alternative pairings. The integrated picture contributes to enabling these costly computations to be performed rapidly in order to understand if a feature with several attributes fits the requirements.

F. Ada Boost Training

In addition to selecting the best features, AdaBoost is also a new idea which trains the classifiers. Although there are about 160.000 functionalities in the 24/24 detector window, only certain of them are crucial to identify a face. So, in the 160,000 function, we apply the AdaBoost algorithm to find the best features. Each hair-like function represents a weak learner in the Viola-Jones algorithm. AdaBoost examines the performance of all classifiers you deliver You assess the classification performance of all the photos used for training in all sub-regions. Some sub-regions have a significant reaction in the classification system. These are classed as positive, implying that the classifier believes it has a human face. In the view of the classifiers, sub-regions that do not produce an effective response do not have a human face. They are categorised as negative. The good performance classifiers are more important or more important. The ultimate outcome is a strong classifier, which incorporates the top performing weak classifiers, also termed an increased classifier.

G. Cascading Classifiers

Training in the Cascade classification calls for both positive and negative imagery. The positive is the thing being discovered, the negative is the thing not being found. It is mostly a question of discarding non-faces and spending more time on likely face areas to lower calculation costs. The Cascade classifier consists of phases with a powerful classifier in each level. The objective of each level is to assess whether or not the supplied sub fenster is a face. Each step in this algorithm thus has a crucial function to play for the fast and reliable facial recognition process.

II. PROBLEM STATEMENT

Since the 1960s, the field of facial recognition has been a subject for study. It was significant both because the issue was practical and because of the cognitive scientists' theoretical curiosity. Face recognition is intended to verify or identify the identification of a person using a single photograph or a video feed of his or her face. Face

recognition systems, such as safety and health care, are used to track patient consumption and help pain management operations properly. In this case Face recognition systems are employed.Researchers lately paid more attention to this topic, conducted numerous experiments and continually improved the existing models. In computer vision, Convolution neural networks (CNNs) are frequently utilised to enhance state-of-the-art for numerous applications. The availability of vast volumes of training data is one of the most significant components.Based on the fact that it is often the case that a face recognizer is built up, especially if the dataset is restricted, it is difficult.One of the main problems with a restricted dataset is that if various lightnings, a person's face may appear different, yet different people may have identical appearances. Assume you should design a mobile ID unlocker. The person wouldn't be able to demand millions of photographs for the facial recognition system to be uploaded. In this circumstance, it would be an appropriate method to use only one or a few samples.

III. LITERATURE REVIEW

Imane et al. [8] presented to the classifier and to the uniform local binary pattern histogram (ULGBPHS) for pattern scan a face detection system employing HOAR cascades, standardisation and emotion detection by utilising CNN on FER 2013(KNN). The model employed 4 distinct machine learning methods, i.e. 70 percent at 106 epochs with the use of KNN and SVM algorithms, and the accuracy rates
(SVM, KnN, random forest, classification, regression trees) were good. This model can be improved with no algorithms for machine languages.
Sepidehsadat et al.[9] suggested that the network's attention on the face should be made simpler by using a Gabor filter since the output orientations are excellent for the facial wrinkles, which will then be an input in the CNN. The network focuses on the useful characteristics with an age accuracy of 7% and sexual accuracy of 2%.
In order to enhance the overall result, Ari Ekmekji[10] has devised a model that combines interrelationships between sex and age. The weaknesses include the complexity of separating the data into folds, training and cross-validating the classifiers and merging the resultant classificators into a test-ready classifier.

IV. OBJECTIVES

The goal of this project is to implement a real-time face recognition system using deep learning. This as it proceeds from the assumption that there is only limited images available to learn from. Algorithms should be evaluated based on accuracy. Humans are capable of determining an individual's age and gender relatively easily using facial attributes. Although it is challenging for machines to perform the same task, in the past decade incredible strides have been made in automatically making prediction from face image. The project identifies or detects the age and gender from the given face images. The tools used involve Convolutional Neural Network along with programming language like Python. The project has been motivated by problems like lack of security, frauds, child molestation, robbery, criminal identification.

V. METHODOLOGY

A. Face Detection with Viola-Jones-Haar Cascades

We start by deploying the OpenCV algorithm for Viola-Jones and recognise faces in real time in the camera feed. All you need is to instal OpenCV and Python on your PC. This is really straightforward.
We have many Haar Cascade models in OpenCV that are trained and stored in XML files. We utilise this file rather than create and train the model from scratch. Haar cascade XML file which is a webcam classifier for the identification of a certain item. Using OpenCV to detect the head face, you have haarcascasde frontalface default.xml. OpenCV links to a camera that users may use to scan their faces for age, sex and emotional categorization.In this work, we will be using the file "haarcascade frontalface alt2.xml." Let's begin coding now. One approach is to locate the route of the file "haarcascade frontalface alt2.xml." We achieve it via the Python language os module.

B. Working of Viola-Jones Classifier for Detecting Faces

For the detection of individual faces, we utilise haarcascade frontalface default.xml. This file has a set of features with a face (eyes, nose, beard,etc). If measurements are big, we will use the picture dimensions to improve the output:
img=cv2.read
print(‘original dimensions:’)
scale_percent=40% of original size

Step 1: First, the path to HaarCascade FrontalFace default.xml has to be found. We utilise a Python language os module and utilise the OpenCV CascadeClassifier function to detect the xml file. The XML file path passes to the OpenCV function of CascadeClassifier().
import os
face_cascade=
cv2.CascadeClassifier('haarcascade_frontalface_alt.xml')

Step 2: After loading the classifier, let us open the webcam using this simple OpenCV one-liner code video_capture = cv2.VideoCapture(0)
Next, we need to get the frames from the webcam stream, we do this using the read() function. We use the infinite loop to get all the frames until the time we want to close the stream.
while True:
# Capture frame-by-frame
ret, frame = video_capture.read() The read() function returns:

- The actual video frame read (one frame on each loop)
- A return code

If we have run out of frames, the return code notifies us that this will happen if we reading from a file. When we read from the camera, this does not important, because we may record it indefinitely, therefore we ignore that fact.

Step 3: For this specific classifier to work, we need to convert the frame into grayscale. gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
The object faceCascade contains a MultiScale() detecting function that gets an argument of a frame(image) and runs the classification cascade over the picture. MultiScale means the algorithm looks at multi-scale picture sub-regions to recognise faces of different sizes.
faces = faceCascade.detectMultiScale(gray,scaleFactor=1.1,minNeighbors=5,minSize=(60, 60), Let us go through these arguments of this function:

1) Scale Factor

Specifies how much each picture scale reduces the size of the picture. You can scale the input image to a smaller one and detect it by means of an algorithm. 1.05 is a decent figure, which suggests that you are resizing by a tiny step.

2) Min Neighbors

parameter indicating how many neighbours should be retained by each candidate rectangle. The identified faces are affected by this parameter. Higher value results in less but higher quality detections. For it, 3~6 is a positive value.
Flags: Operating mode

3) Min Size

Minimum object size possible. Smaller than disregarded objects.
All detections for the target picture are now on the variable faces. Pixel coordinates are stored for detections. The co-ordinates in the top-left corner, the width and height of the rectangle covering the identified face determine each detection.

Step 4: We will put a rectangle over it to show the identified face.
The rectangle() of OpenCV creates pictures across rectangles, and the top-left and top-right corner pixels coordinates need to be known. The co-ordinates display the pixel row and column. These coordinates may be simply obtained from the variable face.
We also know the position of the face, we define a new region including a person's face and call it the face of the ROI. We identify and round the eyeballs with the circular function on the ROI face.
for (x,y,w,h) in faces:cv2.rectangle(frame, (x, y), (x + w, y + h),(0,255,0), 2)
 # Get Face
face_img = image[y:y+h, h:h+w].copy()
The function rectangle() accepts the following arguments:
The original picture
the top-left detection point co-ordinates
the lower-right detection point co-ordinates
The rectangle colour (a tuple that determines the quantity of the red, green, and blue (0.-255), the following parameters are supported by rectangle().
The thickness of the rectangular lines is just 255 in our instance and we set the green part to be nil.

Step 5: Then, the frame will be only displayed and a means out of this endless loop and close the video stream will also be set. Pressing the 'q' button allows us to escape here.
cv2.imshow('Video', frame)
if cv2.waitKey(1) & 0xFF == ord('q'): break
The next two lines are just to clean up and release the picture. video_capture.release()
cv2.destroyAllWindows()

4) Classification using Convolutional Neural Network

The coagulating layer is the basic building block of CNN. Convolution is a mathematical technique which combines two data sets. In our scenario, the input data is converted to a feature map using a convolution filter.A convolutional neural network consists of several layers. Implicit explanation about each of these layers is given below.

5) Convolution Layer (Conv Layer)

The Conv layer is the core building block of a Convolutional Neural Network. The primary purpose of Conv layer is to extract features from the input image.

6) Pooling Layer (Sub-sampling or Down-sampling)

Pooling layer decreases the sizes of maps by employing functions such as the average or maximum value to summarise the sub-regions. The pooling process is done by sliding over the input a window and passing the window contents into a pooling function.The goal of pooling the network is to minimise the number of parameters (so-called sampling) and increase the robustness of the learnt functions, making them even more reliable on changes in scale and orientation.

7) ReLU Layer

ReLU is a non-linear operation, which represents the Rectified Linear Unit. ReLU is a pixel-based procedure which replaces all negative pixel values in the character map with zero. The goal of ReLU is to add nonlinearity on our ConvNet, as we would like our ConvNet to learn most of the real world data is nonlinear. Instead of ReLU other non-linear functions such as tanh or sigmoid can be utilised, however in most circumstances, ReLU is superior.
Output = Max(zero, Input)

8) Fully Connected Layer

The fully connected layer is just what its name implies: it is completely linked to the previous layer's output. All neurons in the preceding layer (be it completely interconnected, pooling or convolution) are connected to each fully connected layer and are linked to each neuron. Adding a completely linked layer is also a cost-effective technique to learn non-linear combinations. Most characteristics learnt through convolution and pooling layers may be useful, but these characteristics may be much better together.

9) Softmax

The topmost level of the proposed architecture is a softmax layer that calculates the optimum loss term

during training and also the probability for the category during classification. While certain loss layers such as the multi-class SVM loss deal with the output of a fully connected layer as class scores, softmax treat these scores, also termed the multinomial logistic regression, because of the classes' unexpected log chances.

10) Flow of steps

The flow are the steps for the proposed method
Step 1: Load the dataset
Step 2: Detection of Face using Haar Cascade Classifier
It is the main and crucial element of any image of the face. In certain cases, a picture could include rather than face the distinct items. Thus it is crucial to detect the face in this respect[43]. In this research, due of its quickest detection property, the Viola-Jones face detection algorithm is employed. The characteristics of the skin of the face picture are necessary instead of other portions of the photos.

Step 3: Crop the Detected Face Image and Resize It
First, the photos are rescaled to 256 x 256 before the input is applied to CNN and the network will be supplied with the crop 227 by 227. Before extracting the features, each picture input will be scaled to meet the CNN input format.

Step 4: Feature Extraction
The resized input is sent for the function extraction phase during this step, which extracts the features using the CNN[11]. The network architecture provides knowledge about the layers, filters and FC layers employed.

Step 5: Testing using Neural Network
Caffe is utilised for the classification of age and gender of humans in which the values of retrieved characteristics above CNN layers are utilised for caffe and Keras. Here, though, coffee is only for categorisation.

VI. IMPLEMENTATION

A. Training a CNN using Caffe

In training a CNN with coffee there are 4 steps:
Step 1: Preparation of data: This step allows us to clean and save the pictures in a manner that Caffe can utilise. We will write a Python script to handle both pre- and storage images.
Step 2: Definition of Model: In this phase, we select a CNN architecture and describe its parameters in a.prototxt file.
Step 3: Definition of Solver: The model optimization solver is responsible. In a configuration file with extension .prototxt we define solver parameters.
Step 4: Training Model: We train the template from the terminal with one Caffe command. We will obtain the trained model in a file with extension.caffemodel when the model has been trained.
After the training phase, we will perform forecasts on new unknown data using the learned model
.caffemodel. We are going to develop a script for Python.

B. Gender Detection with CNN

The predicted gender may be one of ‘Male’ and ‘Female’ The forecast sex may be one of "man" and "female" CNN age sensing: Age
The output level of CNN in that CNN includes 8 values for 8 age ranges of - (0-2), (4-6), (8-12), (15-20), (25-32), (48-53), (60-100), and is a class of 8 values.
OpenCV provides a method for deep learning classification picture preprocessing:

- blobFromImage ().
- Mean substraction
- Scaling
- And swaping of the canal optionally.

This way blobFromImage builds an image blob in four dimensions. Resize and crop the central picture, remove the average value, the scaling scale values, switch Blue and Red channels.
blob = cv2.dnn.blobFromImage(image, scalefactor=1.0, size, mean, swapRB=True)
Image: here is the pre-processed input image before it's sent through our deep neural classification network.
Size factor: We may choose scale our pictures by a factor whenever we conduct a mean subtraction. This default value is 1.0 (i.e. no scaling), however another value can also be supplied.
Size: Here we are providing the spatial size expected of the CNN.
Mean: they are the mean value of our subtraction. They can be 3 times the amount of RGB, or they can be a single number, in which case each channel of the image is removed. If the mean subtraction is performed, ensure that 3fold is provided in (R, G, B) order, particularly when using SwapRB=default True's behaviour.

```
blob = cv2.dnn.blobFromImage(face_img, 1, (227, 227), MODEL_MEAN_VALUES, swapRB=False)
#Predict Gender gender_net.setInput(blob)
gender_preds = gender_net.forward()
gender = gender_list[gender_preds[0].argmax()] #Predict Age
age_net.setInput(blob) age_preds = age_net.forward()
age = age_list[age_preds[0].argmax()]
```

VII. RESULTS AND OBSERVATIONS

A. Experimental Settings

Because of a high number of parameters, CNN requires plenty of training data. Furthermore, training is extremely time-consuming, optimization may need hours or months. In order to solve this obstacle, two stages are used to develop a transfer learning strategy:
Prior to training: randomly initialised networks are initially trained by an accompanying task which has sufficient pictures labelled.
Fine-tune step: settings which have been learned during pre-training are utilised to begin a new job.

B. Results

The input can be either taken from the webcam or the images can be given as the input. Here, the input image is taken from webcam. The age and gender is found. This method achieves high to medium accuracy. For testing we take sample images

estimating age and gender. For classification we use Caffe model Figure 5.1 and Fig 5.2 represents the sample images with correct age and gender classifications. Figure 5.3 represents the pictures of age misclassifications. Figure 5.4 represents the pictures of gender misclassifications.

Results of using Caffe Deep Learning Framework
If the detected face is a male, the output is M. If the detected face is a female, the output is W.
For the age prediction, the CNN's output layer(probability layer) in this CNN consists of 8 values for 8
age classes of the following ranges- (0 – 2), (4 – 6), (8 – 12), (15 – 20), (25 – 32), (38 – 43), (48 – 53),
(60 – 100).

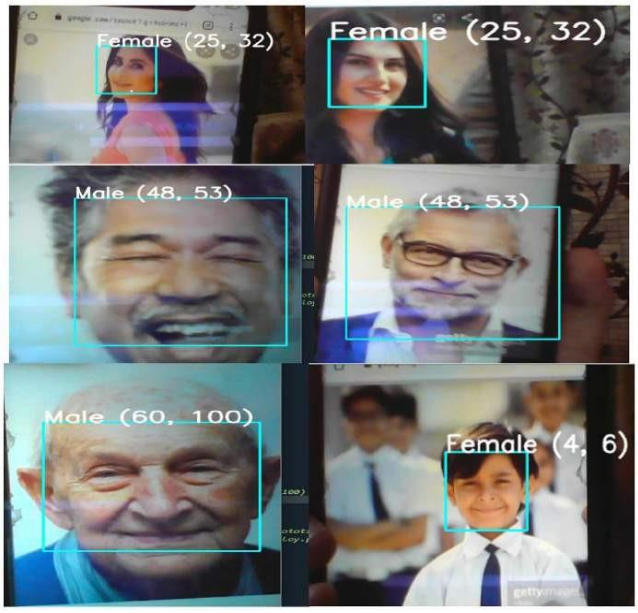

Figure 1: Samples of face images with correct age and gender classifications

Figure 2: Samples of Age misclassifications

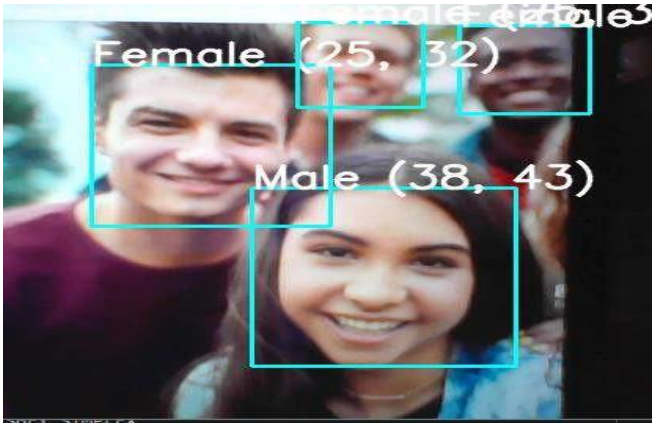

Figure 3 : Multiple Human Age and Gender Prediction

Caffe Model	Exact Match	
Males Total: 50	38	76%
Females Total: 50	40	80%
Total : 100	78	78%

Table1: Caffe Model Accuracy

C. 7.2 Rate of Classification/Accuracy

Accuracy = No. of accurate prediction Total no. of prediction

The overall accuracy of the system using caffe deep learning framework is 78%.

VIII. CONCLUSION

We assess the design of the CNN for good performance in this project. Age estimates and gender estimates via the convolutive Neural Network are the suggested approach of this research. The pre-trained CNN model was utilised to extract the features from the picture. Accuracy was 76 percent correspondingly in the results analysis for age and gender predictions using the caffe model. The model was designed in python language. Real time and static detection of the face have been performed. In a single picture, the system can recognise numerous faces

REFERENCES

[1]. Transactions on Pattern Analysis and Machine Intelligence, p. 1, 2019. E. Agustsson, R. Timofte, S. Escalera, X. Baro, I. Guyon, and R. Rothe, "Apparent and real age estimation in still images with deep residual regressors on appa-real database," in Proceedings of the 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), pp. 87–94, Biometrics Wild, Bwild, Washington, DC, USA, June 2017.

K. Zhang, C. Gao, L. Guo et al., "Age group and

gender estimation in the wild with deep RoR architecture," IEEE Access, vol. 5, pp. 22492–22503, 2017.

[3]. A. Kuehlkamp, "Age estimation from face images," in Proceedings of the 6th IAPR International Conference on Biometrics (ICB), pp. 1–10, Madrid, Spain, June 2013.

[4]. V. Carletti, A. S. Greco, G. Percannella, M. Vento, and I. Fellow, "Age from faces in the deep learning revolution," IEEE

B. Bin Gao, H. Y. Zhou, J. Wu, and X. Geng, "Age estimation using expectation of label distribution learning," in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, pp. 712–718, Stockholm, Sweden, July 2018.

R. C. Malli, M. Aygun, and H. K. Ekenel, "Apparent age estimation using ensemble of deep learning models," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 714–721, Las Vegas, NV, USA, June 2016.

[7]. G. Antipov, M. Baccouche, S. A. Berrani, and J. L. Dugelay, "Apparent age estimation from face images combining general and children-specialized deep learning models," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 801–809, Las Vegas,

NV, USA, June 2016.

[9]. G. Antipov, M. Baccouche, S. A. Berrani, and J. L. Dugelay, "Effective training of convolutional neural networks for face-based gender and age prediction," Pattern Recognition, vol. 72, pp. 15–26, 2017.

R. Rothe, R. Timofte, and L. Van Gool, "Deep expectation of real and apparent age from a single image without facial landmarks," International Journal of Computer Vision, vol. 126, no. 2–4, pp. 144–157, 2018.

[11]. H. Han and A. K. Jain, "Age, gender and race estimation from unconstrained face images," Tech. Rep., Michigan State University, East Lansing, MI, USA, 2014, MSU Technical Report, MSU-CSE-14-5.

[12]. J. Huang, B. Li, J. Zhu, and J. Chen, "Age classification with deep learning face representation," Multimedia Tools and Applications, vol. 76, no. 19, pp. 20231–20247, 2017.\

[13]. E. Eidinger, R. Enbar, and T. Hassner, "Age and gender estimation of unfiltered faces," IEEE Transactions on Information Forensics and Security, vol. 9, no. 12, pp. 2170–2179, 2014.

[14]. Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In Proc.

[15]. Conf. Comput. Vision Pattern Recognition, pages 1891–1898. IEEE, 2014

[16]. E. Eidinger, R. Enbar, and T. Hassner. Age and gender estimation of unfiltered faces. Trans. on Inform.

[17]. Forensics and Security, 9(12), 2014