# Applied Computer Vision: Identify Age of Person

Janvier Nshimyumukiza
Carnegie Mellon University Africa
Kigali, Rwanda
www.cmu.edu

## 1. Abstract

*In this work we were asked to estimate the age of a person in a photo using facial image. What I did is to implement the code in C++ that would receive an image or a sequence of images and predict his/her image using facial features. I have implemented this solution using 4 different methods that I will elaborate on in the next paragraphs. Those methods perform differently in increasing order of accuracy from method 1 to method 4. This means that method 1 is least accurate model and method 4 generate most accurate solution for the problem.*

## 2. Tools Used

To do this task, I have used C++ 11 programming language and OpenCV version 4.4.

### 2.1. What is OpenCV?

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

## 3. Identifying Age of Person

Identifying age of a person based on the facial features is among the most expensive task in deep learning. Achieving high accuracy for this task requires several days or even weeks of model training. This task is complex because eve human is most likely to make bad prediction. The human's prediction can even be 15 years far from the real age especially for old people.

My solutions are basically based on instantiations of pretrained caffemodel. This model has been trained on the IMDB-WIKI dataset of over 500 thousand images. This model has relatively good performance but it can have an error of up to 10 years. However, in my methods, I have applied several interpolation techniques to adjust the model's output and including linear interpolation, neighborhood interpolation, and adjustable neighborhood interpolation.

Before elaborating more on my solution methods, I will first talk about deploying the caffemodel into work.

### 3.1. Face detection with Haar Cascade Classifier

Face detection is very important in building and deploying age estimation model. Like I have said before, this is one of the complex tasks to we don't want to learn useless features. In my model

deployment, I have used to pass in it with face detected from the input image. In this work is that for every input image I call my defined face detection model to crop the face in the image and if it not there the image will not be passed into the model. To do this task, I have used opencv's cascade classifier.

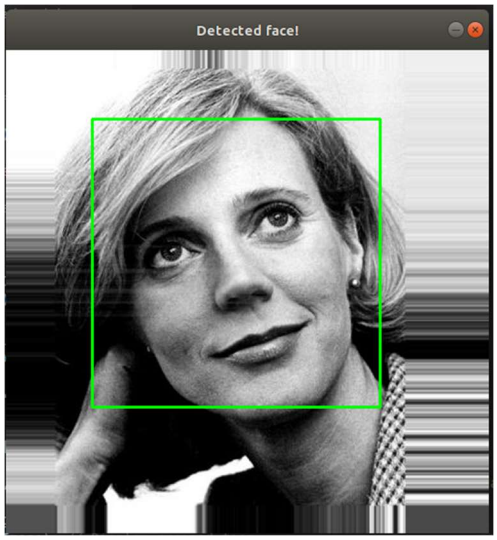


**Figure 1 Using Cascade Classifier, I have detected the face in the image**

Now after detecting the face, I have extracted the face from the image and resize the face to be 224x224 the standard size for the most CNN models.

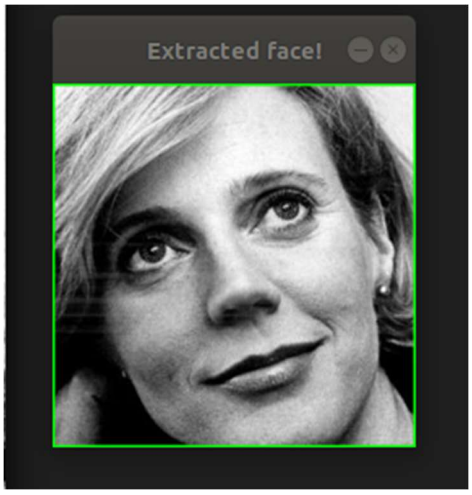


**Figure 2 The extracted face was resize to the standard input size.**

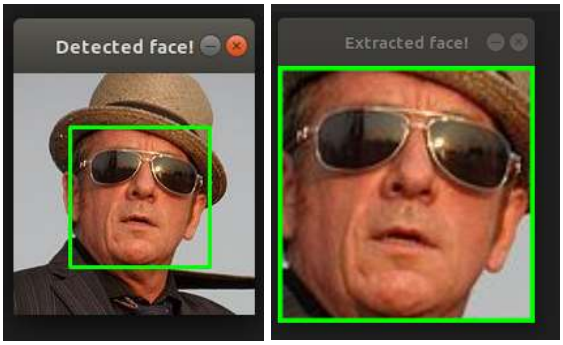


**Figure 2 The original image(left) was smaller than the standard input and so the extracted face(right) has to be resized to match the standards.**

### 3.2. Age Prediction

Now using the caffemodel that I have loaded using cv::dnn::readNetFromCaffe, I have mode the prediction and it is the straight forward method. The model outputs are in a vector of probabilities that the age of a person belongs to a given class of age, this is because the model uses softmax activation function which returns probabilities distributed among classes. This model has 101 classes which refer to age classes of 0, 1…100.

### 3.3. Alternative Working Solution Methods

I have implemented four methods to get the best out the probability vector. The following paragraphs elaborate on each of the method.

#### Method1: Argmax

In this method I have selected the index with maximum likelihood. This refers to the most probable age class that the person might be belonging into. Yes, this technique gives the result, but it can be improved.

**Method2: Linear Interpolation of the probabilities with the class**

In this solution, using the probabilities as the weight scores, I have then calculated the age using linear interpolation techniques.
Let's say that the vectors:

$$C = < 0, 1, 2, \dots 100 > \qquad and$$
$$P = < p_0, p_1, p_2, \dots p_{100} >$$

P and C are the vectors of age classes and the probabilities that age of a given person belong in a certain class. In this method I have calculated the age of a given person as

$$Age = \sum_{i=0}^{100} p_i * C_i \qquad where\ C_i = i$$

Ci is the class. This would give linearly interpolated solution.

**Method 3: Neighborhood Interpolation**

Critically, the probability that a person of 60 years old belong to a class of 2 or 10 years should be considered irrelevant. This probability score shouldn't matter for a large difference of years. Therefore, I have thought of doing linearly interpolation on the neighborhood with greater likelihood. So, since it likely to get an error of at least $\pm 5$ I have decided to do the interpolation of the 11 indices centered at the index of position with maximum probability score. This improves the solution of the methods 1 by exploiting the neighboring classes. This is actually because if the model predicts the age of a person to be 50 there a significant probability that person's age is between 45 and 55. Interpolating this interval would give more precise estimation of age.

**Method 4: Intervals**

In this method, I have first considered the interval 0-2, 3-6, 7-12, 12-17, 17 – 22, 22-29, 29 – 37, 37 – 47, 47 – 59, 59 – 75, and 75 – 100.
Classifying the age of a person in intervals and then after use intervals to do the final estimation have outperformed all the other methods.
As you can see, the interval boundaries grow as the as the ages increases and I have done this for a reason. It is easier to confuse the people aged 70 years old and 90 years old than confusing people aged 10 and 20. Growing intervals have also shown the improvement of the estimation accuracy.

### 3.4. Challenges and incomplete Alternative Methods

In this assignment, it would be suitable to use Residual Network to do this task. ResNet are the complex architectures made up of stacks of several Convolutional Neural Network layers. Their complexity has shown good performance in tasks that require advanced features extraction like face classification, age estimation, …
I was motivated to use resnet34(my working python code submitted) and resnet50 for this task and I have implemented the running code. However, due to the complexity of the training process, I couldn't finish training model due to limited computational resources and time.
Inconsistent availability of the cloud resources was also a challenge. For example, I have tried to train resnet34 using Google Colab Pro but this was still not enough to train this model. I have also tried training resnet50 using AWS but this was very expensive to do it.

### 3.5. Sample results

For the image in figure 1:

Real age: 27
Method1: 21
Method2: 23
Method3: 22
Method4: 28

For the image in figure 58:

Real age: 58

Method1: 52
Method2: 46
Method3: 51
Method4: 51

- END