Age Detection Image Processing: An approach

Luis Miguel Garcia Marin
Fulda University of Applied Science
luis-miguel.garcia-marin@informatik.hs-fulda.de

Contents

ABSTRACT

In this paper we introduce ourselves into the understanding of how we try to guess the age of a person just by looking at them and how this was tried to be implemented with different approaches of Machine Learning. We also see the results of my own approach with Convolutional Neural Networks (CNN), whose code it is provided in the references [1].

1. INTRODUCTION

For sure you have asked before how old could be that person that you met the other day in the cafe or your favourite place and who took all your attention. We usually look at wrinkles in the skin, the texture and color of the hair or other facial or body features to try to guess a person's age. It could surprise us in how many features do our brain focus to try to guess this (even in their emotions [2]), and it was also studied in deep by many researchers [3].

1.1 Starting to solve it

Traditionally, detecting the age and gender of a person given an image of him or she has been an exciting task to solve along years. Initially, the problem was tried to be solved with traditional Machine Learning techniques [4] and with the principals that humans usually follows (like focusing in wrinkles in the skin, color of the hair...) as we mentioned before. But this approach did not get very successful results, since modelling all of this features was definitely not an easy task and they had to do many simplifications.

1.2 Our approach

I propose an approach with modern Deep Learning techniques, using the library TensorFlow [5] in Python [6] to build a model with enough successful results. The goal is not to build the best Neural Network for this purpose, but to give my own try and see the results.

2. STATE OF THE ART

Nowadays we can see a lot of work related to the age detection, since the improvement of the Machine Learning techniques and how the Deep Learning has revolutionized it. But some ages before, were also related work to this field using traditional Machine Learning techniques, as we mentioned before [4]. Steven E. Campana, M. Christina Annand and James I. McMillan tried to determine the consistency with Graphical and Statistical Methods [7]. But one of the first works that presented practical computations for visual age classification from facial images was in 1999 by Kwon, Young H and da Vitoria Lobo and Niels [8].

Some decades after, everyone can make their own model for age detection with their own Personal Computer and Deep Learning tools. For example, Prerak Agarwal [9] tried to solve this problem using traditional Machine Learning (using models like RandomForestClassifier or SVC) and Deep Learning techniques and compared the results, having a validation accuracy of 83.0% with Convolutional Neural Networks while with SVC the best result was of 53.4%. These results mean that the improvement in the use of CNN is clear.

3. METHODOLOGY

3.1 Getting the data

Before starting building the CNN model that we are going to train, we need the main ingredient in this recipe, which can not be other than the data. Some people try to make

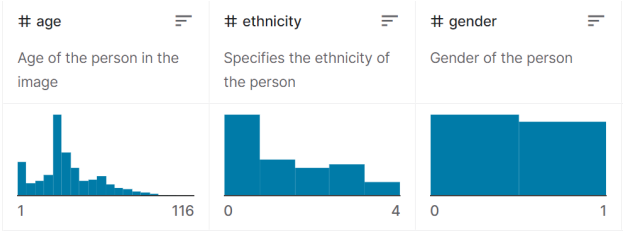

Figure 1. Amount of images of the dataset classified by age, gender and ethnicity.

their own data of images and labels, taking a big amount of facial photos and assign to all of them a label, but this is a really tedious task. Therefore, I am going to use a dataset that I found in Kaggle.com [10], from Nipun Arora. This streamlines my workflow considerably.

This dataset provided by Nipun Arora includes 27305 facial images of 48x48 pixels from people aged between 1 and 93 years, and they are labeled on the basis of age, gender and ethnicity. This means that it can be used for many more purposes, but in our case this is more than enough. We can see the amount of images that we have classified by age, gender and ethnicity in the Figure 1.

3.2 Technologies, tools and libraries

As I mentioned before, for this approach I am going to use TensorFlow [5], a Google end-to-end machine learning platform, and Python [6], a programming language very commonly used for machine learning projects. But this is not enough, we need some more libraries: Numpy and Pandas for correctly loading, managing and operating dataframes (or tensors) of the dataset, Matplotlib for showing plots and graphics of our data, Keras to get the necessary layers to build our model, and Sklearn to use its "Train Test Split" function, in order to split our dataset into train and validation datasets.

3.3 Converting the pixels of the data to a Numpy array

Once we load the dataset of 27305 facial images of 48x48 pixels with their labels, in our case in .csv format, using Pandas and we explore the data, we see that the pixels are expressed as strings separated by spaces. To better handle this data, we are going to convert it into an array of numbers, with the help of a lambda function, which uses the functions x.split() (to separate the elements by each space) and np.array() (to build the array, with 32 precision float number type). The result of this is that we can see now that the pixels are a numerical array.

3.4 Visualizing the data

After this conversion, we are now able to preview some images of the dataset. I am going to show about 20 images, using the plot function of Matplotlib and accompanied by all the labels. However, in the next steps I will only take the 'age' label, since it is the only one that it is useful for this purpose. A visualization of this can be seen in the Figure 2.

Figure 2. Visualization of 20 images of the dataset with their labels.

3.5 Processing the images

Extracting the images: We extract the pixels in a variable ("x"), converting them into a tuple to be able to correctly access the .shape attribute. Normalizing the images: We normalize the pixels so that the model can work better with floating values between 0 and 1. Knowing that the maximum value of a pixel is 255. Converting pixels from 1D to 3D: In order to have information about the nearby pixels and to be able to perform convolution, we now reshape the pixels to go from working with one dimension to three dimensions (width in pixels, height in pixels and number of color channels). In this way, the input of our neural network will also have these dimensions.

3.6 Extracting the labels and spliting the datasets

We get in another variable ("y") the age labels. Since there is no predefined total number of classes (the maximum age of a person does not have a strict limit), I am not going to categorize them. After this, we are ready to split the datasets of images ("x") and labels ("y") into their train and test (also called validation) datasets respectively, and to do this, we use the "Train Test Split" function of the library Sklearn.

3.7 Building the model

We build the neural network model, in this case it is a modification of the one on the notebook 3 (Convolutional Neural Networks) of the Nvidia Deep Learning Introduction Course [11], used to detect the number in an image with the MNIST dataset. The input must be 3-dimensional as specified previously (48,48,1). However, we want the output to be a real number that indicates the predicted age of the person in the photo, so the output will be a single unit with Relu activation, since we do not have, as other times, a specific number of categories between which distribute

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 48, 48, 32)        320
_________________________________________________________________
batch_normalization (BatchNo (None, 48, 48, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 24, 24, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 12, 12, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 4608)              0
_________________________________________________________________
dense (Dense)                (None, 64)                294976
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 387,841
Trainable params: 387,777
Non-trainable params: 64
_________________________________________________________________
```

Figure 3. Summary of the Convolutional Neural Network model.

the solution percentages. You can see the summarizing of the model in the Figure 3.

3.8 Compiling and training the model

In the compilation of the model we indicate to use 'adam' (method of the descent of the gradient stochastic) as an optimizer, the root mean square error as a loss function and the error mean absolute as metrics. Then, we proceed to carry out the training of the model, with 20 epochs.

4. RESULTS

As we can see from the Figure 4, we end up obtaining loss value of 90.9845, but the most important value is the validation loss of 92.0247, which not differs so much from the training loss value, so we can confirm that there is no over-fitting in this train. We can also say that those values are not the best ones, but they are not bad ones either. This means that they are good enough to possibly have decent results, but the best way to see this is to test it with some examples.

Now, I create a make predictions function to make separate predictions and test the model. And then I finally create the age detector function, which makes use of the make predictions function and presents the result to us in a more legible way. I am going to use this function with 2 image examples: Princess Leonor's image (15 years old) and an old woman's image (90 years old). The results of these 2 examples can be seen in the Figure 5.

As we can see from this 2 experiments, the neural network predicts better the age of a younger person (the error is only of 1 year) than of an older person (the error in the second example is of 90 - 52 = 38 years), but what is this due to? This can be explained by looking at the amount of images of the used dataset classified by age (Figure 1).

```
Epoch 1/20
289/289 [==============================] - 3s 11ms/step - loss: 332.3342 - mae: 13.8695 - val_loss: 1088.2957 - val_mae: 27.6129
Epoch 2/20
289/289 [==============================] - 2s 8ms/step - loss: 202.7516 - mae: 10.6614 - val_loss: 587.2354 - val_mae: 19.2056
Epoch 3/20
289/289 [==============================] - 2s 7ms/step - loss: 168.1027 - mae: 9.6140 - val_loss: 122.9288 - val_mae: 8.0803
Epoch 4/20
289/289 [==============================] - 2s 7ms/step - loss: 153.4476 - mae: 9.1409 - val_loss: 173.9724 - val_mae: 9.6985
Epoch 5/20
289/289 [==============================] - 2s 7ms/step - loss: 145.0757 - mae: 8.8778 - val_loss: 102.8419 - val_mae: 7.6934
Epoch 6/20
289/289 [==============================] - 2s 7ms/step - loss: 139.2640 - mae: 8.6717 - val_loss: 94.8595 - val_mae: 7.1141
Epoch 7/20
289/289 [==============================] - 2s 7ms/step - loss: 131.8873 - mae: 8.4012 - val_loss: 114.1211 - val_mae: 8.1974
Epoch 8/20
289/289 [==============================] - 2s 7ms/step - loss: 128.5595 - mae: 8.3177 - val_loss: 141.6526 - val_mae: 8.3377
Epoch 9/20
289/289 [==============================] - 2s 7ms/step - loss: 126.5166 - mae: 8.2660 - val_loss: 99.5473 - val_mae: 7.2386
Epoch 10/20
289/289 [==============================] - 2s 7ms/step - loss: 117.4219 - mae: 7.9854 - val_loss: 145.8851 - val_mae: 8.5983
Epoch 11/20
289/289 [==============================] - 2s 7ms/step - loss: 117.0364 - mae: 7.9370 - val_loss: 89.2123 - val_mae: 6.9273
Epoch 12/20
289/289 [==============================] - 2s 7ms/step - loss: 111.1664 - mae: 7.7419 - val_loss: 160.5373 - val_mae: 9.1037
Epoch 13/20
289/289 [==============================] - 2s 7ms/step - loss: 109.5348 - mae: 7.6836 - val_loss: 155.7063 - val_mae: 8.7982
Epoch 14/20
289/289 [==============================] - 2s 7ms/step - loss: 107.5804 - mae: 7.5717 - val_loss: 127.6717 - val_mae: 7.9955
Epoch 15/20
289/289 [==============================] - 2s 7ms/step - loss: 104.4058 - mae: 7.5231 - val_loss: 158.4935 - val_mae: 8.9667
Epoch 16/20
289/289 [==============================] - 2s 7ms/step - loss: 100.9138 - mae: 7.3601 - val_loss: 92.0247 - val_mae: 7.1669
Epoch 17/20
289/289 [==============================] - 2s 7ms/step - loss: 93.3130 - mae: 7.1371 - val_loss: 97.9601 - val_mae: 7.2063
Epoch 18/20
289/289 [==============================] - 2s 7ms/step - loss: 94.6301 - mae: 7.0824 - val_loss: 99.4097 - val_mae: 7.4861
Epoch 19/20
289/289 [==============================] - 2s 7ms/step - loss: 88.6848 - mae: 6.9435 - val_loss: 93.8473 - val_mae: 7.2125
Epoch 20/20
289/289 [==============================] - 2s 7ms/step - loss: 90.9845 - mae: 7.0288 - val_loss: 92.1674 - val_mae: 7.0196
```

Figure 4. Training results of the Convolutional Neural Network model.

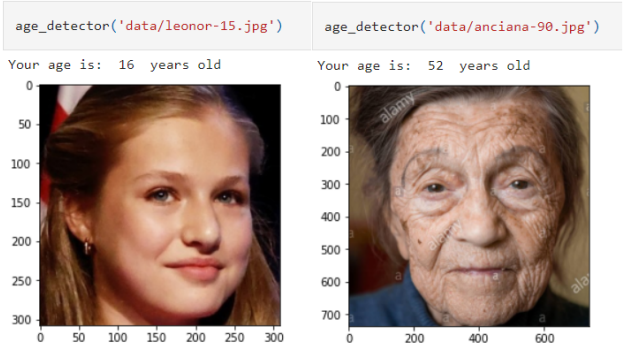

Figure 5. Testing the Convolutional Neural Network model with 2 image examples.

We can see that there are more samples of persons with a young age than of older ages like 90 years old persons. Therefore, the neural network is not so trained for this situation. This could be solved by increasing the amount of the dataset by adding more images of elderly people.

5. DISCUSSION

An approach to build a model for detecting the age of a person given their facial image was introduced and this lead to some ways in which this model could be improved and get better results.

The future work could be increasing the variety of the dataset and making some changes in the layers of the model and test it in a heuristic way. In addition, I could try to expand the responsibilities of the model to predict also the gender and the ethnicity of a person based on their facial image.

But for the moment, this is good enough for experimentation and learning how to manage the Neural Network development. This is what it is behind the "Artificial Intelligence" which everyone is talking about, and it is not magic, it is just some adjustments and calculus in order to minimize the value of a loss function, as we have seen before.

Special thanks to Nvidia Deep Learning Institute Courses

[11], from which I have learned many of the concepts and techniques presented here.

6. REFERENCES

[1] L. M. G. Marín, "Image Recogniser for the Detection of a Person's Age," https://github.com/lintenn/age_detector, 2021, [Online; accessed 6-December-2022].

[2] C. S. Monk, E. B. McClure, E. E. Nelson, E. Zarahn, R. M. Bilder, E. Leibenluft, D. S. Charney, M. Ernst, and D. S. Pine, "Adolescent immaturity in attention-related brain engagement to emotional facial expressions," *Neuroimage*, vol. 20, no. 1, pp. 420–428, 2003.

[3] E. H. Aylward, J. Park, K. Field, A. Parsons, T. L. Richards, S. C. Cramer, and A. N. Meltzoff, "Brain activation during face perception: evidence of a developmental change," *Journal of cognitive neuroscience*, vol. 17, no. 2, pp. 308–319, 2005.

[4] J. Wallinga, "On the detection of osl age overestimation using single-aliquot techniques." *Geochronometria: Journal on Methods & Applications of Absolute Chronology*, vol. 21, 2002.

[5] Google, "TensorFlow - An end-to-end machine learning platform," https://www.tensorflow.org, 2022, [Online; accessed 11-December-2022].

[6] P. S. Foundation, "Python Programming Language," https://www.python.org, 2022, [Online; accessed 11-December-2022].

[7] S. E. Campana, M. C. Annand, and J. I. McMillan, "Graphical and statistical methods for determining the consistency of age determinations," *Transactions of the American fisheries Society*, vol. 124, no. 1, pp. 131–138, 1995.

[8] Y. H. Kwon and N. da Vitoria Lobo, "Age classification from facial images," *Computer vision and image understanding*, vol. 74, no. 1, pp. 1–21, 1999.

[9] P. Agarwal, "Age Detection using Facial Images: traditional Machine Learning vs. Deep Learning," https://towardsdatascience.com/age-detection-using-facial-images-traditional-machine-learning-vs-deep-learning-2437b2feeab2, 2020, [Online; accessed 11-December-2022].

[10] N. Arora, "AGE, GENDER AND ETHNICITY (FACE DATA) CSV," https://www.kaggle.com/datasets/nipunarora8/age-gender-and-ethnicity-face-data-csv, 2020, [Online; accessed 11-December-2022].

[11] N. Corporation, "Getting Started with Deep Learning," https://courses.nvidia.com/courses/course-v1:DLI+S-FX-01+V1/, 2022, [Online; accessed 15-December-2022].