ADAM MICKIEWICZ UNIVERSITY in POZNAŃ
FACULTY OF SOCIAL SCIENCES
Kognitywistyka

Gracjan Popiółkowski

Algorithms for Recognizing People's Age from Facial Photos

Poznań 2023

Contents

Introduction

In this report, I present a comparison of the results of classifying facial images into four age groups: 6 - 20 years, 25 - 30 years, 42 - 48 years, and 60 - 98 years. I will analyze various models with different parameters in terms of effectiveness and the problem of overfitting. Additionally, I will verify the performance of my models against one of the popular models, i.e., VGG19.

CHAPTER 1

Data Presentation

1.1. Dataset

I used the "Age Recognition Dataset" provided by Rashik Rahman on the Kaggle platform. The dataset contains over 9,000 images, classified into 4 age groups. The images are face crops with various ethnic features, not excluding background, and have a resolution of 200x200 pixels.

1.2. Data Preparation

All images were resized to 128x128 pixels. I used the 'ImageDataGenerator' from the Keras library to dynamically generate augmented training data during the learning process. The generator was configured to:

- Scale pixel values from 0-255 to 0-1.
- Randomly flip images horizontally and vertically.
- Rotate images within a range of -20 to 20 degrees.
- Reserve 20% of the data for validation purposes.

All of this was aimed at increasing the diversity of the training data.

CHAPTER 2

Image Classification Algorithms

Image classification algorithms are a type of machine learning models that learn to recognize and assign labels to images based on patterns and features present in those images. Image classification algorithms are widely used in various fields, such as object recognition, medical diagnostics, satellite image analysis, and many others. A popular tool in this domain is convolutional neural networks (CNN) due to their ability to effectively detect patterns in multidimensional space. In my analysis, I focused on modifying three key parameters essential for image classification:

1. `Number of filters in convolutional layers` - influences the detection of more complex features in the images.
2. `Size of densely connected layers` - increasing the number of neurons in densely connected layers can help the model better fit the training data.
3. `Dropout` - a regularization technique that randomly "turns off" certain neurons during training, which can help prevent the model from overfitting.

`VGG19` is one of the CNN models designed for image classification. The name "VGG19" refers to VGGNet, which stands for Visual Geometry Group Net, and the number "19" indicates the number of layers in the network, including 16 convolutional layers and 3 fully connected layers. The VGG19 model was originally trained on the large ImageNet dataset, which comprises millions of images across thousands of categories.

In the following sections, I will present a comparison of the classification results after changing the mentioned parameters and compare these results with those of VGG19.

CHAPTER 3

Results Analysis

3.1. The Simplest Neural Network Model

`Model 1` – this will be the simplest of my models, and it will serve as a starting point. Its specification:

- `Conv2D Layer` – in this layer, I applied 16 filters with a size of 3x3, activated by the ReLU function. This is a key component of convolutional neural networks, responsible for feature extraction from the image.
- `MaxPooling2D Layer` – this layer uses max pooling on the results of the previous convolution. It reduces the dimensions of the matrix while maintaining the key features of the image.
- `BatchNormalization` – applied after each convolutional layer and the Dense layer. It helps speed up the learning process by normalizing activations in the layer, ensuring stability during learning.
- `Flatten Layer` – used to transform the multidimensional matrix into a one-dimensional vector, which is required before moving to the Dense layers.
- `Dense Layers` – the model contains three Dense layers. The first has 128 neurons, the second has 32 neurons, and the third is the output layer with 4 neurons (corresponding to the number of categories). The activation in the densely connected layers is ReLU, except for the output layer, where softmax is used for classification.
- `Adam Optimizer` – an adaptive gradient optimization algorithm used to minimize the loss function. Adam is popular due to its efficiency and flexibility.
- `Dropout` – the model does not include the Dropout function, so it may be prone to overfitting.

I will compare the models based on `accuracy` and `val_accuracy`, which are variables that assess classification performance on the training and validation (test) datasets. `Val_accuracy` measures how well the model generalizes to new, previously unseen data. It's important to check whether the model is not overfitting, which can occur when training accuracy is high, but validation accuracy is lower. The results for the baseline model:

- `Accuracy` = 0.7070
- `Val_accuracy` = 0.6461

Thus, the classification accuracy on new data is 64.61%. This is a very good result considering the 4 classification groups, where random assignment would result in 25%. Despite the lack of a Dropout function, the model does not appear to be heavily overfitted, likely due to the small number of filters in the convolutional layers and the number of neurons in the Dense layers.

3.2. Comparison of 3 Models with a Single Parameter Change

The second part of my analysis was conducted by constructing 9 models, which differ from the base version by only one parameter.

3.2.1. Model 2 - Increased Number of Filters in Conv2D

In Model 2, I decided to increase the number of neurons in the hidden layer. This choice aims to enhance the network's ability to detect more complex patterns in the training data. Two layers were used - the first with 32 neurons and the second with 16. The final layer, containing a single neuron with a sigmoid activation function, is used for binary classification. The model was trained for 10 epochs. The performance of this model was:

- `Accuracy` = 0.7070
- `Val_accuracy` = 0.6461

3.2.2. Model 3 - Increased Number of Neurons in Dense Layers

For Model 3, I took the opposite approach, reducing the number of neurons in the hidden layer. A smaller number of neurons can help avoid overfitting, especially with limited training data. The layers of the model consist of 8, 4, and 1 neuron, with a sigmoid activation function. This model was also trained for 10 epochs. The performance of this model was:

- `Accuracy` = 0.7070
- `Val_accuracy` = 0.6461

3.2.3. Model 4 - Adding the Dropout Function

In Model 4, I introduced an additional hidden layer to increase the model's capacity. The added layer has 8 neurons and uses the ReLU activation function. The model consists of three layers: two hidden layers and one output layer with a sigmoid activation function. It was trained for 10 epochs. The performance of this model was:

- `Accuracy` = 0.7070
- `Val_accuracy` = 0.6461

3.2.4. Comparison of the First 4 Models

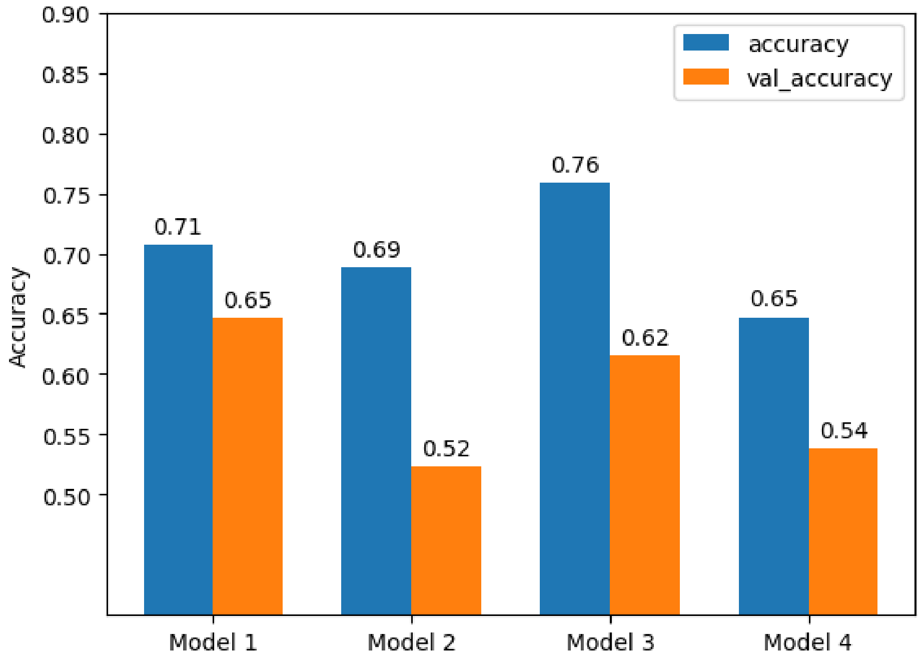

Figure 3.1: The bar chart shows a comparison of the baseline `Model 1` and three modified models. `Model 2` - increased number of filters, `Model 3` - increased number of neurons in Dense layers, `Model 4` - added Dropout function.

Ultimately, we see that `Model 1` performs best in terms of effectiveness on new data. Next, we will see how this changes when we modify more than one parameter in a given model.

3.3. Two Other Combinations and VGG19

In the third part of my analysis, I will examine the results when combining changes to our three main parameters and present the results of the pre-trained VGG19 model.

3.3.1. Model 5

Although Models 2 and 4 did not perform well, I decided to test all three parameter changes in a single model, hoping that the performance would balance out potential overfitting. `Model 5` has an increased number of filters in the convolutional layers to 32, the number of neurons in the Dense layers to 256, 64, and 4, and a Dropout function set to 0.3. The results are as follows:

- `Accuracy` = 0.6652
- `Val_accuracy` = 0.5839

Overfitting is relatively low, but as we can see, the performance is still significantly lower than in the baseline model. I also experimented with changing the Dropout value from 0.3 to both higher and lower values, but the results were still unsatisfactory.

3.3.2. Model 6

Finally, I tested a model where I did not add the Dropout function at all, but I increased both the number of filters in the convolutional layers to 32 and the number of neurons in the Dense layers to 256, 64, and 4. The results are:

- `Accuracy` = 0.7731
- `Val_accuracy` = 0.7056

This result is clearly better than `Model 1`, and to my surprise, the model does not appear to be overfitted. Below is a chart for comparison:

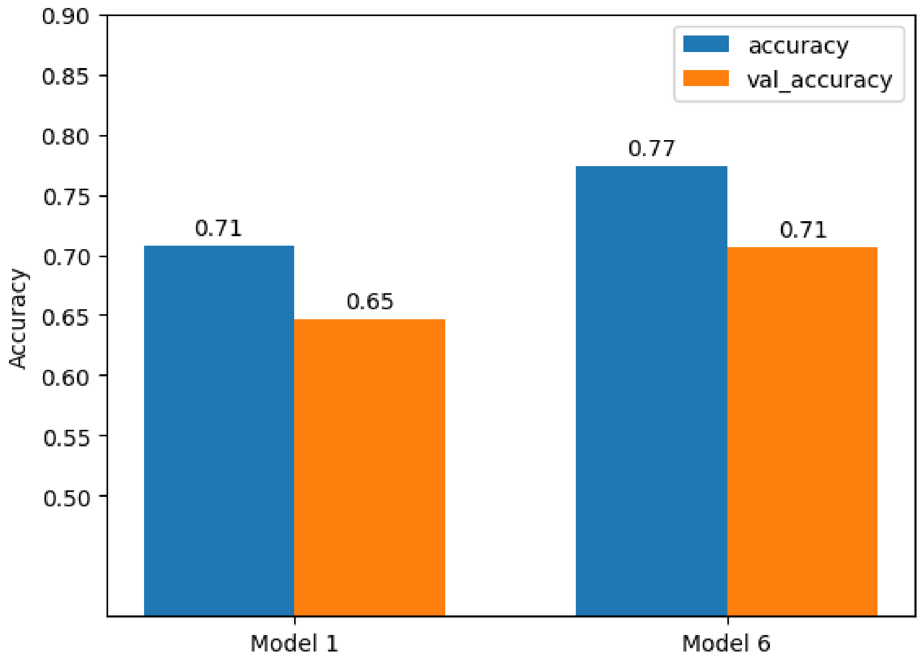

Figure 3.2: Comparison of `Model 1` with the best combination of parameters achieved in `Model 6`.

3.3.3. Model VGG19

`Model VGG19` - the previously described pre-trained model with significant image classification effectiveness. In our task of matching age to facial images, it achieved the following results:

- `Accuracy` = 0.7327
- `Val_accuracy` = 0.6472

It performs slightly better than our baseline model but significantly worse than the best model we managed to achieve.

3.3.4. Chart Comparing 4 Key Models

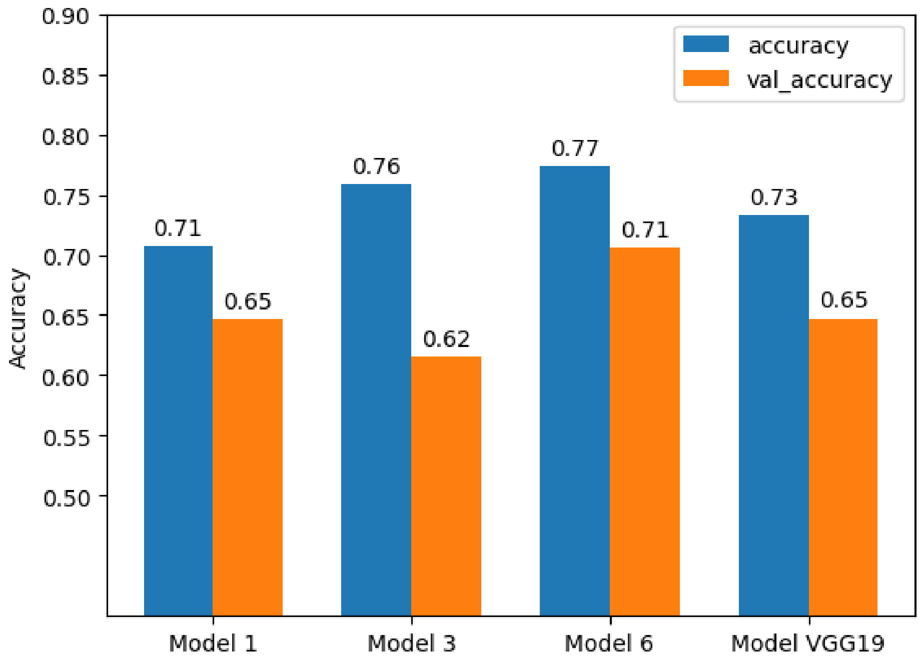

Figure 3.3: Comparison of the best-performing models: `Model 1` - baseline, `Model 3` - with an increased number of Dense neurons, `Model 6` - the best model achieved, and the pre-trained `VGG19 Model`.

Summary

In conclusion, in my work, I used various neural network models to classify facial images into age groups: 6 - 20 years, 25 - 30 years, 42 - 48 years, and 60 - 98 years. The neural networks performed this task remarkably well. The best result I was able to achieve is:

- `Accuracy` = 0.7731
- `Val_accuracy` = 0.7056

Random assignment across 4 groups gives us 25% accuracy. `Model 6` achieved a result of 70% correct classification of the images into the age groups on the test data, which I consider a very good result. These images are not cropped from their background, so they contain a lot of noise from the perspective of an algorithm that analyzes pixel by pixel. They are also low-resolution, and with more powerful hardware, it would be possible to train the algorithms on larger images, i.e., a greater amount of data, from which the models could extract patterns indicative of age.

Bibliography

- https://www.statsoft.pl/textbook/stneunet.html#artificial