Age estimation based on facial images using Deep Learning techniques

Paulo Rijnberg (snr. 2033812), April 2020

1. Introduction

Age estimation plays a significant role in different scenarios, from law enforcements identifying individuals to social interactions. Because the aging process does not only dependents on intrinsic factors (e.g. genetic factors) but also on extrinsic factors (e.g. environment, lifestyle), estimating someone's age based only on their face is a challenging problem[1].

Therefore, the literature offers a wide variety of approaches to this problem. Lots of researcher tried approaching this problem by using traditional Machine Learning techniques such as K-Nearest Neighbors[2] and Support Vector Machines[3]. Since deep learning showed great potential on images, some have proposed working with deep learning techniques. Sithungu and Van der Haar for example focussed on a lightweight model and proposed a modified LeNet-5 architecture[4]. While Ozbula, Aytar and Ekenel researched the potential of using transfer learning with pre-trained model such as AlexNet and VGG1. [5] Lastly, Fariza, Mu'arifin and Arifin also proposed using the ResNe(x)t-50 architecture[6]. Also, most of these proposed models used the imbalanced dataset UTKFace.

For that reason, this paper will focus on the following research question: '*To what extent can we apply deep learning techniques on age estimation using a balanced dataset of facial images?*' Where we take the paper from Sithungu et al. as a starting point, but in our case we used a balanced dataset.

2. Experimental procedure

2.1 Dataset

We used 24,106 *Aligned&Cropped* images from the UTKFace dataset. Then we binned the data into 7 classes, applied under sampling and removed images with ages over 70 (see Table 1). Lastly, we split the data into three sets, 75% training, 20% validation and 5% test.

Table 1: Dataset

Age class	Age range	Samples (imbalanced)	Samples (balanced)
0	1-10	3,492	1,183
1	11-20	1,682	1,183
2	21-30	7,806	1,183
3	31-40	4,345	1,183
4	41-50	2,103	1,183
5	51-60	2,226	1,183
6	61-70	1,183	1,183

2.2 Experimental setup

We started by implementing the (modified) LeNet-5 architecture just like Sithungu et al. We tried modifying these by adding extra layers, changing the activation, optimizer and adding regularization. After this, we turned to transfer learning using architectures as VGG16 and VGG19 (Ozbula et al.) and ResNet-50 (Fariza et al.). We experimented with these by (un)freezing layers, adding extra dense layers and training the model after importing pretrained weights or from scratch. In every experiment we used a generator to load and rescale the data, used batch sizes of 32 and ran each experiment for 15 epochs. We also used both a local machine and Google Colab to run experiments (especially for the ResNets).

2.3 Evaluating the performance

Like mentioned previously, we focused on using a balanced dataset, while most other papers used an imbalanced dataset. Thus, our starting point (the modified LeNet-5) that reached an accuracy of 56% in the paper of Sithungu et al., is not representative in our case. Therefore, we have a baseline of 14% (1,183 /8,281 total balanced samples). We also take into account a confusion matrix and the mean class accuracy on the test set (see GitHub page [7]).

3. Results

Table 2 shows the most interesting results of the experiment. We display here the model architecture, input shape, compiler and highest accuracy achieved in 15 epochs. In notebook *summarized-results.ipynb* (see GitHub page) we also provide the mean class accuracies, confusion matrixes and layer activations (e.g. in Figure 1) for each model on the test set.

Figure 1: Example of activations VGG16 (3) on the test set.

Table 2: Models used in the experiment

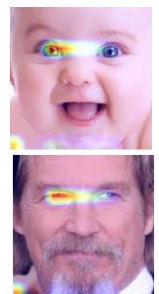| Model architecture	Input shape	Compiler	Acc
1. Original LeNet-5 architecture with L2 regularizer(0.001) in both dense layers.	32, 32, 1	RMSprop, accuracy, categorical cross entropy	0.56
2. Modified LeNet-5 architecture proposed by Sithungu et al.	200, 200, 3		0.56
3. Non-trainable VGG16 layers (weights 'imagenet') with 3 trainable dense layers (512, 512, 7)	200, 200, 3		0.49
4. Non-trainable VGG19 layers (weights 'imagenet') with 3 trainable dense layers (512, 512, 7)	224, 224, 3		0.51
5. Retrain ResNet-50 layers from scratch with 2 extra layers (GlobalAvgPool2D, dense 7)	224, 244, 3	Adam, accuracy and categorical cross entropy	0.51
6. Trainable ResNet-50 layers (weights 'imagenet') with a GlobalAvgPool2D and dense (7) layer	224, 244, 3		0.56

4. Discussion and conclusion

Based on the accuracies in Table 2, mean class accuracies and the confusion matrix, we can conclude that we managed to reach for each model in Table 2 an accuracy greater than the baseline of 14%. Based on those results, we propose model 1 (LeNet-5 architecture with L2 regularization). Please consider that the model had mostly trouble in identifying images from classes 31-40, 41-50 and 51-60 (see confusion matrix on GitHub). We selected the LeNet-5 architecture over the ResNet-50 (model 5) in our scenario because the amount of time needed to make a prediction is significantly smaller due to the amount of layers compared to the ResNet-50. We also found that most ResNet models did show strange layer activations. Nonetheless, we do see potential in using the ResNet architecture when working with a greater dataset.

Like shown in section 2.1 Dataset, we considered under sampling leaving us with *only* 1,183 samples per class. We could argue that these are not enough images to fully (re)train a model (e.g. model 5). Also, because of the limited computation power we only trained our models for 15 epochs. We believe that enhancing the epoch size will lead to overfitting on the training set, but also allows us to apply techniques such as regularization. Even though we were limited to 15 epochs, we still tried regularization for our proposed model. Unfortunately, we did not manage to increase the validation accuracy but we did noticed the validation accuracy showed the most linear trend. We suggest that further work could focus on using a bigger dataset and/or apply image augmentation while keeping the classes balanced, enlarge the epoch size and apply (L2) regularization.

[1] Han, H., Otto, C., & Jain, A. K. (2013). Age estimation from face images: Human vs. machine performance. *2013 International Conference on Biometrics (ICB)*. https://doi.org/10.1109/icb.2013.6613022
[2] Gunay, A., & Nabiyev, V. V. (2008). Automatic age classification with LBP. *2008 23rd International Symposium on Computer and Information Sciences*. https://doi.org/10.1109/iscis.2008.4717926
[3] Guo, G., Guowang Mu, Fu, Y., & Huang, T. S. (2009). Human age estimation using bio-inspired features. *2009 IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2009.5206681
[4] Sithungu, S., & Van der Haar, D. (2019). Real-Time Age Detection Using a Convolutional Neural Network. *Business Information Systems*, 245–256. https://doi.org/10.1007/978-3-030-20482-2_20
[5] Ozbulak, G., Aytar, Y., & Ekenel, H. K. (2016). How Transferable Are CNN-Based Features for Age and Gender Classification? *2016 International Conference of the Biometrics Special Interest Group (BIOSIG)*. https://doi.org/10.1109/biosig.2016.7736925
[6] Fariza, A., Mu'arifin, & Arifin, A. Z. (2019). Age Estimation System Using Deep Residual Network Classification Method. *2019 International Electronics Symposium (IES)*. https://doi.org/10.1109/elecsym.2019.8901521
[7] GitHub page https://github.com/paulorijnberg/deep-learning-age-estimation