Guess The Age Contest: Group 04

Cirillo Benedetto[1], D'Amato Stefano[2], Rastelli Alessando[3], Casaburi Adolfo[4]

[1,2,3,4]**{b.cirillo6, s.damato16, a.rastelli2, a.casaburi35}@studenti.unisa.it**

1. Introduction

Real age estimation is a very challenging problem in the field of facial analysis. This issue is applied across a wide range of sectors, from security, in those applications where should be a tight control on minors' activities, to the business, in order to suggest advertisements according to people's ages, encouraging engagement. Unfortunately, state of the art methods make use of ensembles of DCNNs [1], exploiting huge computational resources, making the net not usable in real scenarios. Moreover, current datasets used in the field of facial age estimation are not so large, or they lack in reliability in the annotation of the age. The LAP 2016 dataset [2] is reliable but only contains 7,591 images. Other datasets, like IMDB-Wiki [3], CACD [4], and UTK contain a lot of faces, but they are annotated with the information based on online web crawling and social networks, therefore, reliability is not guaranteed. The VGG-Face2 MIVIA Age Dataset [5], considered in the contest, consists of 575K faces, divided into 81 classes, with a high degree of imbalance among classes. So what we did first was to find the best classifier among different models, and then build a multi-expert system, composed by an ensamble of three of these "weak" classifiers; we used an RUSBoost algorithm to train it and find the best decision rule, involving the three of them, according to the "degree of expertise" of each one.

2. Description of the method

2.1. General architecture

2.1.1. Single expert

After many attempts, explained in section 4, with the most promising CNNs models, we adopted ResNet50 that represent the best trade-off between performances and response time. We designed the single expert as a classifier with 81 class, taking into account the ordinal ranking between the ages.

This is the architecture:

- Pre-processing layers:
  - Rescaling (with a scale factor 1/255)
  - Random Flip (horizontal and vertical')
  - Random Rotation (with a factor of 0.1)
  - Random Zoom (with a factor of 0.2)
- Backbone:
  - ResNet50 with pre-trained weights on the *ImageNet* datasets.
- Neck:

  - Global Average Pooling layer.
  - Dense layer with 512 neurons and a *ReLu* activation function.
  - Batch Normalization layer.

- Head:
  - Dense layer with 81 neurons and a *Softmax* activation function

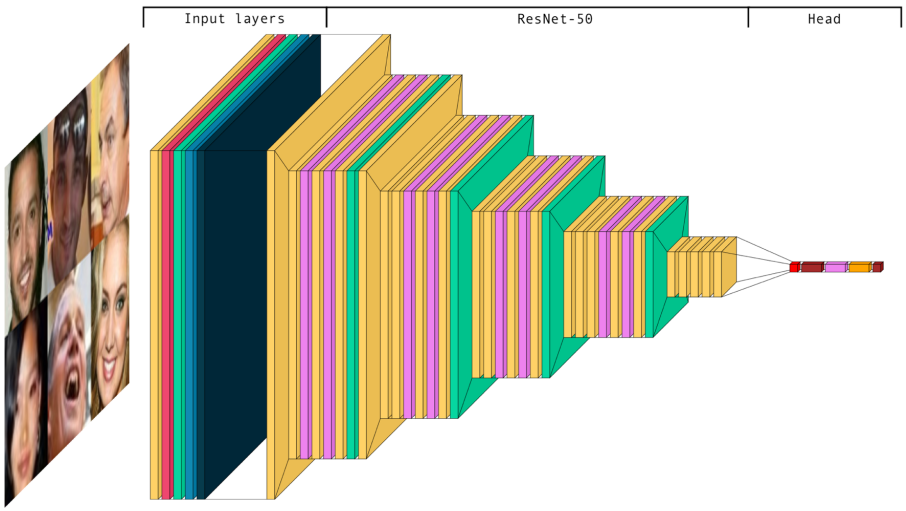

Figure 1. The ResNet-50 structure shown in the figure is for illustrative purposes only

The first layers are useful for data augmentation, as explained in 2.2.3.

The convolutional layers of the ResNet50 are used as backbone, and, in order to perform our custom classification, the last layer is FC with as many neurons as the number of ages in the dataset, that is 81.

The predicted output of the single expert is the expected value of the discrete distribution of the output.

$$\hat{y}(x) = \sum_{k=1}^{81} k \cdot \hat{y}_k(x) \tag{1}$$

where $\hat{y}_k(x)$ is the output probability of each neuron in the FC layer. This choice will be explained in the section 3.

2.1.2. Multi-expert system

Our multi-expert system is composed by $M = 3$ single experts described before. They are trained using the RUSBoost algorithm [7]. RUSBoost combines data random under sampling and boosting techniques. We noticed that training each expert with the whole dataset through a simple boosting algorithm, such as AdaBoost, doesn't improve performance on less represented classes, since imbalance is very high. So we needed somehow to make the imbalance less severe, through over or under-sampling. We used RUSBoost instead of SMOTEBoost, which is algorithm that combines boosting and data over-sampling, since an over-sampling method would have been too expensive and time-consuming on a dataset of our size, despite it would have better performance [8].

Through random under-sampling, we applied a penalization term proportionally to the number of samples of each class in the training set, according to the following rule:

- classes with less then 30% of samples compared to the most represented class → no under-sampling
- class with number of samples between 30% and 65% compared to the most represented class → random under-sampling of 30%
- classes with more then 65% compared to the most represented class → random under-sampling of 50%

Through this configuration, each classifier is trained with a different training set with the following distribution, with respect to the original:

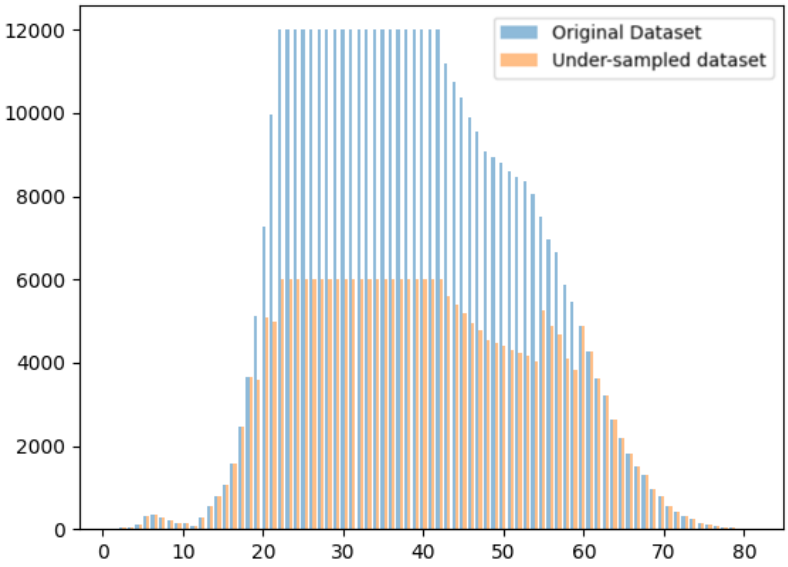

Figure 2. Under-sampling

Since we are dealing with a multi-class classification problem, we adopted and modified the SAMME algorithm [6], instead of using the the traditional AdaBoost algorithm, thought for binary classification problems. Indeed, traditional Adaboost requires that for each weak learner $T^{(m)}$ the error rate must be $err^{(m)} < 1/2$, to obtain the relative $\alpha^{(m)} > 0$ ($\alpha^{(m)}$ is the weight assigned to $T^{(m)}$ in prediction), since the error rate of a random guessing is $1/2$. In fact:

$$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} \tag{2}$$

However, it is much harder to achieve such an error rate in a multi-class classification problem, where the error rate of a random guessing is $err^{(m)} = (1-1/K)$, where K is the number of classes. So the creators of the SAMME algorithm changed the way how alpha is obtained:

$$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} + \log (K - 1) \tag{3}$$

One immediate consequence is that now, in order for (m) to be positive, we only need $(1err(m)) > 1/K$, or the accuracy of each weak classifier to be better than random guessing. Calling $T^{(m)}(x)$ the output of the m-th single expert, the predicted output of the multi expert system is obtained as follows:

$$\hat{y}(x) = \frac{\sum_{m=1}^{M} \alpha^{(m)} \cdot T^{(m)}(x)}{\sum_{m=1}^{M} \alpha^{(m)}} \tag{4}$$

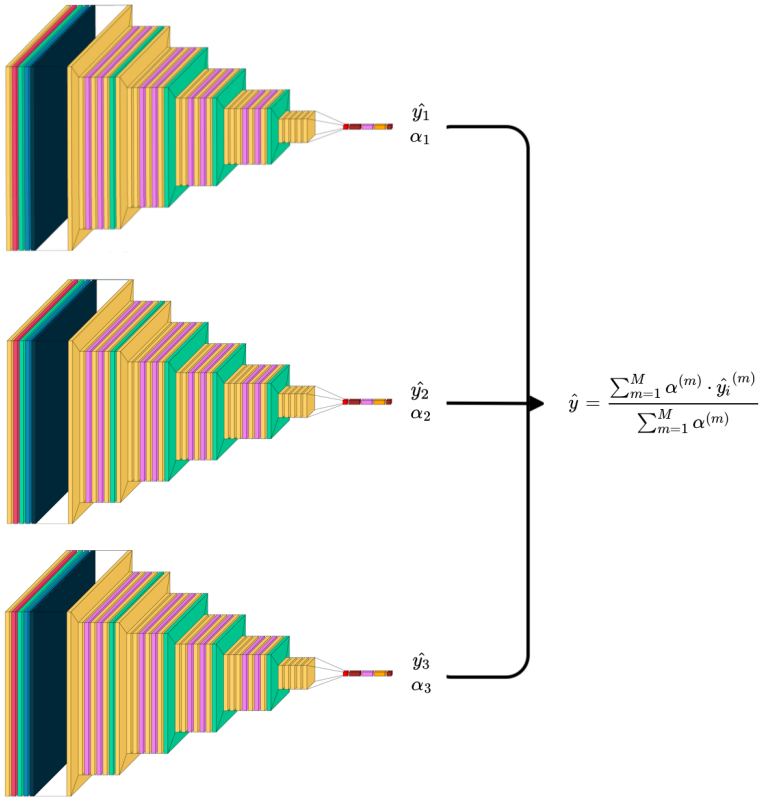

Figure 3. The Multi Expert architecture

2.2. Training procedure

2.2.1. Dataset

The VGG-Face2 MIVIA Age Dataset is composed by 575073 images of more than 9,000 identities. Those images have been extracted from the VGGFace2 dataset and annotated with age by means of a knowledge distillation technique. The 80% of the dataset is used for training, while the remaining 20% is involved in the validation phase. Validation set is not randomly chosen across the whole dataset, but we sampled the 20% from each age class, in order to keep the same distribution among training and validation set. For classes in which the rounded 20% of the number of samples turned out to be 0, we manually inserted a sample in the validation set.

2.2.2. Face pre-processing

Since the faces were already cropped, the pre-processing phase consists only in resizing the input images to 224x224 pixels and rescaling thier values by a factor of 255, in order to keep all values in range 0,1.

2.2.3. Data augmentation

Although the ImageDataGenerator class provides augmentation techniques, the operations are performed on the CPU, slowing down the training. First layers of the network, instead, perform the augmentation as shown in 2.1.1, on the GPU, speeding up the process.

- Random Flip (horizontal and vertical')

- Random Rotation (with a factor of 0.1)
- Random Zoom (with a factor of 0.2)

We performed all these transformations to improve the generalization and robustness of the model. We avoided including certain kinds of augmentations such as noise, blurring, and corruption factors since they could lead to intrusive distortions that would make learning worse.

2.2.4. Training from scratch or fine tuning

Our CNN has been trained starting from pre-trained weights, obtained by training ResNet50 with the dataset ImageNet. All the layers' weights are set to "trainable", making the training longer but more accurate.

2.2.5. Training of the single experts

Each weak classifier is trained over 70 epochs, using early stopping optimization with a patience of 20 epochs, monitoring the loss value on the validation set. We used SGD optimizer with a e base learning rate set to 0.005, which is scaled by a factor of 0.1 after every 10 epochs during which the loss on validation doesn't improve. The default batch size is set to 64. To deal with the problem of class-imbalance we assigned a weight to each sample in the training set for the first weak classifier, according to the ratio between the average number of samples per macro class and the number of samples belonging to the specific macro class. Macro class are groups of ten (or more in the case of the last one) age classes as follows:

I	*II*	*III*	*IV*	*V*	*VI*	*VII*	*VIII*
1-10	11-20	21-30	31-40	41-50	51-60	61-70	70+

Table 1. Age macro class division

We noticed that if weights were assigning according to the 81 age classes and not to the macro classes, some samples would have a very huge weight, making the training too unstable, since imbalance among age classes is very accentuated.

2.2.6. Training of the multi-expert

As described in section 2.1.2, we applied a boosting technique using 3 single experts. We trained the multi-expert system with the following algorithm:

1. *Initialize the samples weights* w_i*, for* $i = 1, ..., n$*, where* n *is the number of samples in the training set.as described in the above section, obtaining distribution* $w0$*.*
2. *Compute* $sum_{w_0} = \sum_{i=1}^{n} w_i$
3. *For* $m = 1$ *to* M*:*
   I. *Fit a classifier* $T^{(m)}(x)$ *to the training data using weights* w_i
   II. *Compute the error rate*

$$err^{(m)} = \frac{\sum_{i=1}^{n} w_i I(c_i \neq T^{(m)}(x_i))}{\sum_{i=1}^{n} w_i} \tag{5}$$

   III. *Compute the alpha values*

$$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} + \log (K - 1) \tag{6}$$

IV. *Update weights, obtaining distribution* $w\{m\}$

$$w_i \leftarrow w_i \cdot \exp(\alpha^{(m)} \cdot I(c_i \neq T^{(m)}(x_i))) \tag{7}$$

V. *Compute*

$$sum_{w_m} = \sum_{i=1}^{n} w_i \tag{8}$$

VI. *Normalize weights to the originary sum*

$$w_i = \frac{w_i}{sum_{w_m}} * sum_{w_0} \tag{9}$$

Distribution of weights for the second and third weak classifiers are calculated according to described algorithm, making the sum of weights always the same and equal to that calculated for the first classifier (such as in fig. 4).

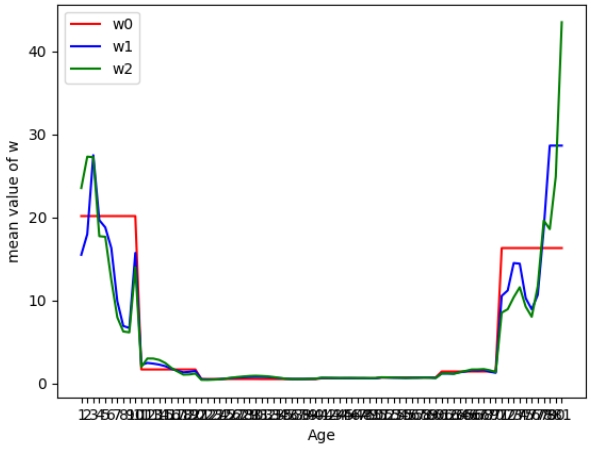

Figure 4. w{i}, with $i \in 0, 1, 2$, is the distribution of weights for the i-the weak classifier

3. Loss function design

To ensure that the predicted age is as close as possible to the true age, while taking into account the competition evaluation metrics, we considered minimizing the following loss:

$$L_{AAR} = L_{mMAE} + L_{\sigma} \tag{10}$$

First define the MAE as:

$$MAE = \frac{\sum_{i=1}^{N} |\hat{y}_i - y_i|}{N} \tag{11}$$

Where N is the number of samples over which MAE is calculated, y_i and $\hat{y}_i$, respectively, the real and predicted age of the i-th element of the batch.

While y_i is read as an integer, $\hat{y}_i$ is the output of a multi-class classifier, as it needs to be converted to an integer, in order to use it in the equation 11. The prediction is computed as the expected value of the discrete distribution of the output (see equation 1) and not as the index of

the neuron with the highest output probability, since any way to do this is not differentiable and it cannot be used to compute the loss.

$$L_{mMAE} = \frac{\sum_{j=1}^{8} MAE^j}{S} \tag{12}$$

Where MAE^j is the MAE computed over the samples belonging to the j-th of the 8 macro-classes in Table 1, and S is the number of macro-classes for which $MAE^j \neq 0$. We didn't divided for the total number of the age macro-classes (8) since batches, over which loss is computed, may not contain sample from each macro class, and this would have generated a unfair value for L_{mMAE}. Loss takes into account a regularization term too:

$$L_{\sigma} = \sqrt{\frac{\sum_{j=1}^{S}(MAE^j - MAE)^2}{S}} \tag{13}$$

that makes errors balanced among macro-classes.

4. Experimental results

4.1. Experimental framework

As described before, dataset is splitted in training set and validation set with a ratio of 80% and 20%. We evaluated different models comparing them through the AAR metric, defined as:

$$AAR = max\{0, 5 - mMAE\} + max\{0, 5 - \sigma\} \tag{14}$$

$$mMAE = \frac{\sum_{j=1}^{8} MAE^j}{8} \tag{15}$$

$$\sigma = \sqrt{\frac{\sum_{j=1}^{S}(MAE^j - MAE)^2}{8}} \tag{16}$$

the MAE metric is defined in equation 11. The designated single experts is the one that minimizes the AAR .

4.2. Results

As can be seen from Table 2 and 3 (the results reported on the different models were obtained with the same hyperparameters described above) the 2 best single models are ResNet50 and its evolution ResNet50V2, but the first was has the best AAR. In these table last letter represents the dataset on which the networks were trained ("I" for ImageNet and "V" for VGGface).

Model	AAR	$mMAE$	σ	MAE
ResNet50_I	**6.815**	2.704	0.481	2.474
ResNet50_V2_I	6.621	2.629	0.750	2.446
ResNet50_V	6.353	3.037	0.610	2.840
SeNet50_V	5.2229	3.714	1.063	3.231
ResNet50_I$_{(regression)}$	5.889	3.331	0.780	3.082

Table 2. Results of different single experts

Model	MAE^1	MAE^2	MAE^3	MAE^4	MAE^5	MAE^6	MAE^7	MAE^8
ResNet50_I	3.683	2.592	2.253	2.473	2.670	2.486	2.458	3.014
ResNet50_V	4.492	2.644	2.563	2.930	3.033	2.947	2.652	3.035
ResNet50_V2_I	4.495	2.365	2.260	2.589	2.499	2.440	2.422	1.960
SeNet50-V	5.906	3.132	2.964	3.417	3.330	3.154	3.251	4.561
ResNet50_$I_{regression}$	5.236	3.068	2.863	3.259	3.245	2.950	2.800	3.228

Table 3. Values of the Mae of the single experts relative to the 8 macro-classes described in Table 1

These results confirm the thesis that the imbalance in the data brought a strong focus on samples of the most represented ages.

As can be seen from Tables 4 and 5, the multi-expert system RUSBoost is the one that give the best results. The difference between the multi-expert systems AdaBoost and RUSBoost (both based on the SAMME algorithm) is that each of the three classifier of the first one are trained with the whole training set, while the second one's classifiers are trained with the under-sampled dataset, as explained in 2.1.2.

Model	AAR	$MMAE$	σ	MAE
AdaBoost	6.825	2.479	0.696	2.250
RUSBoost	**6.918**	2.504	0.577	2.300

Table 4. Results of the multi experts systems compared to the best single expert

Model	MAE^1	MAE^2	MAE^3	MAE^4	MAE^5	MAE^6	MAE^7	MAE^8
AdaBoost	4.492	2.644	2.563	2.930	3.033	2.947	2.652	3.035
RUSBoost	3.882	2.301	2.342	2.404	2.257	2.123	2.125	2.598

Table 5. MAE relative to the 8 macro-classes described in Table 1, achieved by the multi experts systems and the best single expert

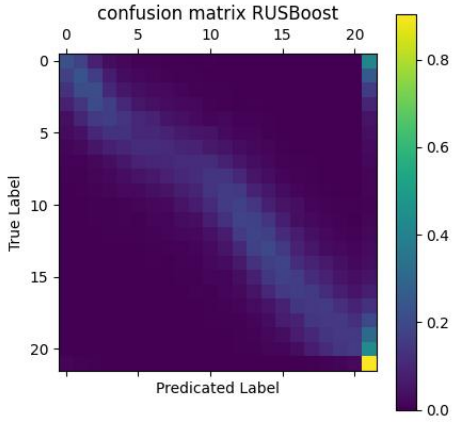

Figure 5. Confusion matrix for the most representative classes

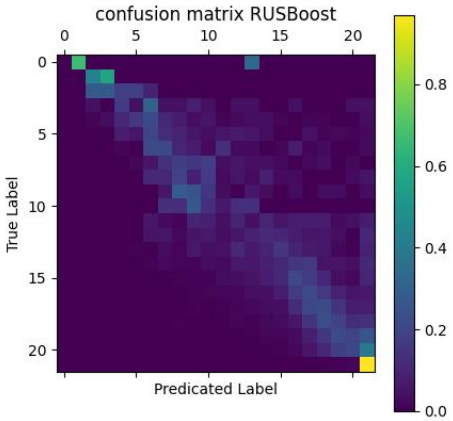

Figure 6. Confusion matrix for the left tail

We report confusion matrices (figure 5 and 6) for the most representative classes (from ages 22 to 42), and for the less representative classes in the "left tail" of the dataset distribution (from ages 1 to 20). As it can be seen, samples belonging to "stronger" classes are miss-classified with a low error (around 2 years), while the other ones are easily classified as older: due to the unbalance of the dataset, classification "pulls" towards central classes, which are the most represented. Confusion matrix for the less representative classes in the "right tail" of the dataset distribution, behave specularly to that just described, classifying samples as younger.

4.3. Repeatability and stability of the results

From the several tests performed, we can infer that the results should be in line with what has been described above, although we cannot affirm that since a K-fold cross validation was not performed due to time constraints (a complete training on the multi expert model on average took 2/3 days depending on load of the workstation we could exploit for this contest).

4.4. Prediction of the results on the test

Assuming that the distribution of the test set is similar to that of the validation set, it can be assumed that the best prediction that the model can perform in the test phase can be similar to the performance described above. Thus: $AAR \simeq 6.9$, $MMAE \simeq 2.5$, $\sigma \simeq 0.6$, $MAE \simeq 2.3$.

References

[1] Carletti, V., Greco, A., Percannella, G., Vento, M.: Age from faces in the deep learning revolution. IEEE Trans. Pattern Anal. Mach. Intell. 42(9), 2113–2132 (2019)

[2] Escalera, S., et al.: Chalearn looking at people challenge 2014: Dataset and results. In: European conference on computer vision, pp. 459-473 (2014)

[3] Rothe, R., Timofte, R., Van Gool, L.: Deep expectation of real and apparent age from a single image without facial landmarks. Int. J. Comput. Vis. 126(2), 144–157 (2018)

[4] Chen, B.-C., Chen, C.-S., Hsu, W.H.: Cross-age reference coding for age-invariant face recognition and retrieval. In: European Conference on Computer Vision, pp. 768–783 (2014)

[5] Greco, A., Saggese, A., Vento, M., Vigilante, V.: Effective training of convolutional neural networks for age estimation based on knowledge distillation. Neural Comput. Appl. (2021)

[6] Ji Zhu, Hui Zou, Saharon Rosset and Trevor Hastie, “Multi-class AdaBoost,” 2009

[7] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse and A. Napolitano, "RUSBoost: Improving classification performance when training data is skewed," 2008 19th International Conference on Pattern Recognition, Tampa, FL, USA, 2008, pp. 1-4, doi: 10.1109/ICPR.2008.4761297

[8] Tanha, J., Abdi, Y., Samadi, N. et al. Boosting methods for multi-class imbalanced data classification: an experimental review. J Big Data 7, 70 (2020)