COMPUTER VISION (2023)

REPORT: **Task 2**

**Group members:**

*Leonardo Bocchi, <lbocchbo37@alumnes.ub.edu>, LeonardoBocchi*

*Jordi Segura Pons, <jsegurpo8@alumnes.ub.edu>, JordiSegura*

**1. SUMMARY OF CONTRIBUTIONS**

The model architecture we adopted makes use of the EfficientNet B3 model, with pre-training on ImageNet. We chose this model because it is particularly accurate while still being very efficient and it is easily accessible through tensorflow. EfficientNet-B3 has been shown to perform well on the Adience benchmark dataset for age estimation from facial images. EfficientNet models are known for their ability to achieve high accuracy with fewer parameters than other models. In particular, it uses a total of 10,783,535 parameters (compared to the 23,561,152 of ResNet) and the same input form as the one we are using.

To refine our previous results, we designed a novel approach that captures the proportion of samples within specific age, ethnicity, expressivity, and sex groups. We divided the ages into 10-year brackets, resulting in 240 unique categories (10x3x4x2). We then proceeded to define our strategy considering the dataset as divided in those cross-feature groups. We worked towards giving more relevance to samples belonging to underrepresented classes and less significance to those from overrepresented classes making use of the following techniques

* Sets of weights intended to create an inverse relationship between the number of samples in a group and the weight on the evaluation of the loss function to such samples
* Class Balanced loss functions (with both implemented and custom losses)
* Optimization of biases during training: Custom loss using the bias scores when evaluated and custom callback to readapt the weights each epoch based on the evaluated bias scores

***1.2. Custom loss strategy (in the case of task 2 or 3)***

We began by using a weighted Mean Squared Error loss function and testing different sets of weights.

For each sample, we assigned a weight based on the prevalence of its particular group in the dataset, aiming to balance each subgroup during the training stage. We then proceeded by testing different types of losses combined with the best sets of weights, aiming to define a suitable Class Balanced loss (such as a Class Balanced Focal loss) that would obtain good results with the distribution of samples in the 240 cross-features groups we defined. We thus tried implementing Effective Number of Samples weights, as they have previously shown good results as techniques of bias mitigation when applied to unbalanced datasets. Finally, we tried including the optimization of the bias scores in the minimization of the loss function through a custom loss and a custom callback to readapt the weights during training.

It needs to be noted that bias mitigation strategies are case-specific and strictly depend on the distribution of the samples in the dataset. Furthermore, our baseline model (the model trained with no data augmentation and no custom loss) produced considerably good results, proving difficult to be improved for some aspects. (e.g. Expression Bias, MAE)

Finally, the custom loss we adopted as the best result of our experiments is a simple weighted Mean Squared Error loss function, making use of a set of weights defined as follows

***1.3. Training strategy***

The adopted training strategy consisted in the training of the whole model (EfficientNet and classifier) in one stage, making use of two training sessions, using a learning rate of 1e-3 and of 1e-4 respectively. The used optimizer is Adam and the training made use of an early stopping criteria with a patience parameter of 5. This strategy has been adopted after different experiments, which have been detailed in the previous report. (Task 1)

**2. EXPERIMENTS AND RESULTS**

We started by evaluating the results of our model making use of the weights proposed in the starting kit. We then proceeded trying other sets of weights computed as different functions of the number of samples or proportions of our groups. In these first experiments we used a weighted mean squared error function as the loss function in order to assess purely which weights could be more effective. (The considered transformations are detailed in the footnotes)

**Table 1: Results of the experiments using different weights**

| ***Model*** | ***Gender bias*** | ***Expression bias*** | ***Ethnicity bias*** | ***Age bias*** | ***MAE*** | ***Bias score*** | ***Bias+MAE score*** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Baseline* | 0.363686 | *0.073896* | *0.390308* | *3.188383* | *4.847096* | *4.016273* | *8.863369* |
| *Sample* | *0.276278* | *0.396767* | *0.140017* | *3.582565* | *5.275298* | *4.395627* | *9.670925* |
| *1.1[[1]](#footnote-0)* | *0.099239* | *0.460264* | *0.338204* | *4.115105* | *5.815720* | *5.012812* | *10.828532* |
| *1.2[[2]](#footnote-1)* | *0.097882* | *0.317216* | *0.365985* | *4.598607* | *5.836170* | *5.37969* | *11.21586* |
| *1.3[[3]](#footnote-2)* | *0.309343* | *0.310098* | *0.817502* | *4.406461* | *5.223919* | *5.843404* | *11.067323* |
| *1.4[[4]](#footnote-3)* | *0.078337* | *0.654515* | *0.468230* | *2.608116* | *6.919300* | *3.809198* | *10.728498* |
| *1.5[[5]](#footnote-4)* | *0.516866* | *0.395914* | *0.096084* | *3.909363* | *5.388470* | *4.918227* | *10.306697* |
| *1.6[[6]](#footnote-5)* | *0.374354* | *0.214759* | *0.175823* | *4.102623* | *5.098659* | *4.867559* | *9.966218* |
| *1.7[[7]](#footnote-6)* | *0.344722* | *0.161563* | *0.201967* | *4.040752* | *5.241138* | *4.749004* | *9.990142* |
| *1.8[[8]](#footnote-7)* | *0.319824* | *0.321396* | *0.345878* | *3.580311* | *6.065234* | *4.567409* | *10.632643* |

The results obtained did not considerably improve our baseline, in particular, only one of them achieved a better bias score as a whole, losing some accuracy. The experiment from which we expected the best results was the one making use of a Class Balanced loss, using Effective Number of Samples (ENS) weights, since in previous work it has proved to be an effective approach.

However, in our case it did not achieve good results, but we decided to further experiment using ENS weights paired with different loss functions.

The best result was obtained using an analogous transformation to the one used in the starting kit, but making use of our cross-features groups. Therefore, we decided to test other loss measures, using the best performing weights and the ENS weights, to try to further improve the results.

**Table 2: Results of the experiments using different loss functions**

| ***Model*** | ***Gender bias*** | ***Expression bias*** | ***Ethnicity bias*** | ***Age bias*** | ***MAE*** | ***Bias score*** | ***Bias+MAE score*** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Baseline* | 0.363686 | *0.073896* | *0.390308* | *3.188383* | *4.847096* | *4.016273* | *8.863369* |
| *1.44* | *0.078337* | *0.654515* | *0.468230* | *2.608116* | *6.919300* | *3.809198* | *10.728498* |
| *2.1[[9]](#footnote-8)* | *0.349664* | *0.266409* | *0.291128* | *4.627850* | *5.511846* | *5.535051* | *11.046897* |
| *2.2[[10]](#footnote-9)* | *0.336481* | *0.753706* | *1.288953* | *3.276047* | *6.912511* | *5.655187* | *12.567698* |
| *2.3[[11]](#footnote-10)* | *0.220882* | *1.358601* | *0.758582* | *3.634614* | *7.497971* | *5.972679* | *13.47065* |
| *2.4[[12]](#footnote-11)* | *0.000779* | *0.296819* | *0.909930* | *5.483409* | *5.701625* | *6.690937* | *12.392562* |
| *2.5[[13]](#footnote-12)* | *0.376532* | *0.249136* | *0.495564* | *3.846174* | *5.490967* | *4.967406* | *10.458373* |

Contrary to what we expected, the Effective Number of Sample set of weights did not produce any improvement when combined with the use of different loss functions. Similarly, the combination of the weights that produced the best result in combination with the Mean Squared Error loss function, when paired with other loss functions did not produce better results on the whole line of bias scores.

Finally we decided to try including the bias scores in the custom loss to try to allow the model to optimize for them. However, we encountered some problems implementing such a custom loss using Tensorflow, due to tensor operations and constraints that are in place for the Automatic Differentiation in Keras to function correctly. (specifically it is not possible to pass non-tensor objects, even if they are used only for distinguishing the samples, e.g. the metadata; an extremely customized approach handling also the batches might be a way around this issue) We thus decided tried another approach, adapting the weights during training. We did so using a custom callback that would compute at each epoch the bias score and use it to transform the weights, changing the samples’ relative significance. The transformation used requires the weights to be normalized, otherwise the loss can be unbounded and the model might not be able to train enough. The experiment we performed used the best previous weights (1.4) and the callback we defined changed the weights according to the following

This works assuming that giving more relative weight to samples belonging to underrepresented groups will improve the bias scores. However, the experiment did not produce better results, and it also required quite a longer training time. This is probably due to the necessity of finetuning the callback function and balancing the reweighting function in relation to the bias scores. Other possibilities would be to consider either different transformations of the weights with the bias scores or considering a different kind of normalization method, possibly even a standardization. We will consider further experimentation in this regard during task 3, since we believe the approach of using the bias scores to readapt the weights during training could produce better results. (We decided not to include the results of these last experiments since they showed no improvements)

**3. FINAL REMARKS**

Testing different sets of weights we observe how not all transformations of the density of the sample distribution are effective in lowering the biases scores. Also, as one would expect, when trying to improve the bias scores, the model obtains lower MAE scores. Some sets of weights improved the bias scores, with one set managing to lower the bias scores in its complexity by around 0.22. Our implementation of the Class Balanced Focal loss combined with the Effective Number of Samples set of weights did not manage to improve the results, despite being a quite broadly used technique for bias mitigation in unbalanced datasets. Finally, the last performed experiments, making use of techniques to include the optimization of the bias scores in the loss function, specifically the custom loss taking into account the biases and the weights readaptation during training according to the bias scores, did not perform better than previous approaches, probably requiring more fine tuning.

In conclusion, making use of a simple weighted Mean Squared Error loss and the cross-features groups we managed to slightly improve the bias scores of our baseline.

***Idea for task 3:***

***Compute the weights using not only the number of samples in each group, but also the correlation of all the features***

1. 1.1: Proportional weights to the relative proportions of each group, wi=1-(nsamples,i/N) [↑](#footnote-ref-0)
2. 1.2: Lighter proportional weights, wi=1-0.8(nsamples,i/N) [↑](#footnote-ref-1)
3. 1.3: Heavier proportional weights, wi=1-1.2(nsamples,i/N) [↑](#footnote-ref-2)
4. 1.4: Starting kit transformation but with our considered groups, wi=N/(nclasses\*nsamples,i) [↑](#footnote-ref-3)
5. 1.5: Inverse Number of Samples (INS), wi=1/nsamples,i [↑](#footnote-ref-4)
6. 1.6: Inverse of Squareroot of Number of Samples (ISNS), wi=1/sqrt(nsamples,i) [↑](#footnote-ref-5)
7. 1.7: Effective Number of Samples (ENS), beta=0.9, wi=(1-beta)/(1-beta# of samples of group i) [↑](#footnote-ref-6)
8. 1.8: Effective Number of Samples (ENS), beta=0.99, wi=(1-beta)/(1-beta# of samples of group i) [↑](#footnote-ref-7)
9. 2.1: Best weights (1.4) with Mean Squared Logarithmic Error [↑](#footnote-ref-8)
10. 2.2: Best weights (1.4) with Huber loss [↑](#footnote-ref-9)
11. 2.3: Best weights (1.4) with Logcosh loss [↑](#footnote-ref-10)
12. 2.4: Effective Number of Samples (ENS) weights, beta=0.9 with Focal loss (Class Balanced Focal loss) [↑](#footnote-ref-11)
13. 2.5: Effective Number of Samples (ENS) weights, beta=0.9 with Mean Squared Logarithmic Error loss [↑](#footnote-ref-12)