OPT 2 – COMPUTER VISION (2022 / UB)
REPORT: **Task 2**

Group members:

Adam Dudek, adudekdu38@alumnes.ub.edu, aa.dudek5
Mert Mecit, mmecitme7@alumnes.ub.edu, mertmecit

1. SUMMARY OF CONTRIBUTIONS

For the age regression problem, we have developed a ResNet101 model. We added new layers to make it suitable for our purpose based on validation errors we get from experiments. We initialized with ImageNet weights and first trained newly added layers, then the full model. After hyper-parameter tuning (batch size, learning rate, and optimizer), we chose a loss function and weight for the loss function. We tried several combinations of them and we believe mean absolute percentage error with gender weights did the best as it has the lowest MSE (our priority) and face bias and also did a good job on other biases as well. We managed to improve 4 out of 5 metrics when compared to what we have done with data augmentation.

2. EXPERIMENTAL SETUP & DISCUSSION OF THE RESULTS

In this part of the task of age regression, we are required to improve our model accuracy and decrease biases on age, ethnicity, gender, and facial expression via custom loss functions. First, we created our model, chose a proper training strategy, and fine-tuned hyperparameters. Then, we considered different custom loss functions and weighted samples to improve our model.

a. Backbone Selection

i. Backbone Modification to Model

For our purpose, we tried three different backbones: ResNet50, ResNet101, and ResNet152. Before comparing our models, we saw that our models were yielding very bad results as they are. They need to be adjusted to our problem. We considered different handcrafted scenarios. We added some additional layers at the end. We used them all on ResNet50 because the models are somewhat similar. We think scenarios are pretty self-explanatory in table 1. Executing them is pretty much taking similar times, so we only considered validation loss. When we add 2 FC layers with ReLU, and 1 FC layer with sigmoid, it yielded the best result.

Table 1: Backbone Modification Selection

Model	Validation Loss
ResNet50 as it is	0.088
ResNet50 + 2 FC layer with ReLU act. + 1 FC layer with Sigmoid act.	**0.017**
ResNet50 + 2 FC layer with ReLU act. + 1 FC layer with Tanh act.	0.019
ResNet50 + 3 FC layer with ReLU act. + 1 FC layer with Sigmoid act.	0.018
ResNet50 + 1 FC layer with ReLU act. + 1 FC layer with Sigmoid act.	0.021
ResNet50 + 1 FC layer with Sigmoid act.	0.024

ii. Backbone Modification to Model

After modifying them, to compare the 3 backbone's performances, we used the same hyperparameters and the same training method for them. 1e-5 of the learning rate, Adam optimizer, 32 batch size, mean squared error loss function, and no pre-training (random initialization of weights) is used. We used 50 epochs for each. We compared their performances solely based on validation accuracy and execution time. Table 2 shows the results. Based on the validation errors, we are relatively inconclusive as validation errors are very close. However, as it has almost %33 less time to train, we stick with ResNet101.

Table 2: Backbone Selection

Model	Validation Loss	Execution Time (in mins)
ResNet50	0.017	60
ResNet101	**0.014**	**100**
ResNet152	0.013	150

b. Training Strategy

After deciding on our backbone (ResNet101), it is time to find a proper training strategy. In total, we would like to train our model with 50 epochs. We considered 3 possible training and tried them with the same scenario, 1e-5 of the learning rate, Adam optimizer, 32 batch size, mean squared error loss function:

1. Initialize weights randomly and train the model with 50 epochs.
2. Pre-train the model with **ImageNet** weights, then train the newly added layers with 25 epochs and then the whole model with 25 epochs more.
3. Pre-train the model with **ImageNet** weights, then train the whole model with 50 epochs.

We have already done the 1st training strategy above. Results are given in table 3. The second strategy, pretraining with freeze yielded the best results as it has a similar validation loss to pretraining without freeze but it is significantly lower in time. This will constitute our training strategy.

Table 3: Backbone Selection

Training Strategy	Validation Loss	Time (in mins)
1) No Pretraining	0.014	100
2) Pretraining with Freeze	**0.010**	**75**
3) Pretraining without Freeze	0.010	100

c. Hyperparameter Selection

We will use ResNet101 with modification of the last layers above and pretrain it with ImageNet weights and the first train added layers, then the whole model. Now it is time to fine-tune our hyperparameters. So far, we have been using 1e-5 of the learning rate, Adam optimizer, 32 batch size, and mean squared error loss function. The loss function will be considered in a separate section. Other than that, learning rate, optimizer and batch size will be discussed here. We used 3 different scenarios for each and used 50 epochs for them. We compared their validation loss and execution time. ADAM optimizer strictly dominates the other optimizes. With batch sizes, 32 and 16 are almost similar both in performance and time, so we tried both 32 and 16 in our next models. With a learning rate of 1-e4, the model is early stopped after 15 epochs. However, 1e-5 improved the model even further in 50 epochs and we prefer to stick to it.

Table 4,5,6: Learning Rate, Batch Size, and Optimizer Fine-Tuning

LR	V. Loss	Time	Batch	V. Loss	Time	Optimizer	V. Loss	Time
1e-5	**0.010**	**75**	**32**	**0.010**	**75**	**ADAM**	**0.010**	**75**
1e-4	0.016*	22	0.010	0.010	75	SGD	0.21	75
1e-6	0.021	75	0.016	0.016	75	AdaDelta	0.22	75

Before proceeding to our loss function and weighted samples optimizations, we would like to present our model and its performance as it is now. Our model is ResNet101 with adjusted layers. We used 50 epochs with ImageNet pretraining. We only trained added layers for the 25 epochs and then trained the whole model with 25 epochs more. Hyperparameters used are 1e-5 learning rate, batch size of 32, and ADAM optimizer. Results of the second leg are given in figure 1 below (in total it has 50 epochs) and summarized in table 7 below.

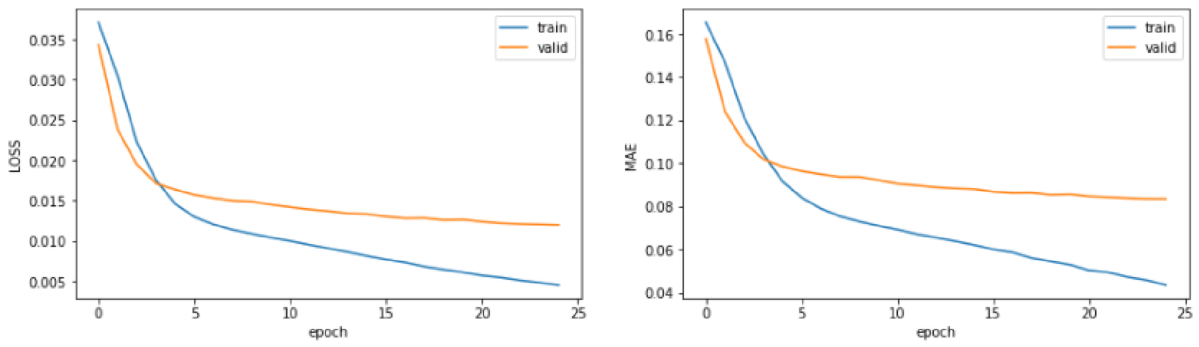

Figure 1: Validation Error Before Loss Function Optimization

Table 7: Accuracy and Bias scores of our Model so far

Model	***Learning rate***	***Training strategy***	***Gender bias***	***Expression bias***	***Ethnicity bias***	***Age bias***	***MAE***
ResNet101 (before Loss Function. Optimization)	*1e-5*	*2 (see part 2-b)*	0.949	1.664	1.546	10.17	10.106

d. Loss Function

Now, we will define our cost function. We used 7 loss functions:

1) MSE: widely utilized loss function in literature.
2) Sparse Categorical Cross Entropy: although it is widely used for classification purposes, we would like to see how it performs in our task.
3) Poisson: also utilized for regression when modeling count data. We also wanted to see how it performs in our model.
4) Huber: it is used for robust regression, not much sensitive to outliers.
5) LogCosh: it combines the advantages of absolute error and mean square error loss functions.
6) Mean Squared Logarithmic Error
7) Mean Square Percentage Error

We used them on our model above. We used both 16 and 32 as our batch sizes and compared the results based on both validation loss and biases.

Our final results show that for different purposes, we can utilize a different type of error function. However, we believe, table 8 shows that the best results are with mean absolute percentage error and logcosh. Mean absolute percentage error yields the best MAE. Which we believe is the most important metric (and the one we prioritize). On the other hand, logcosh does very well on different kinds of biases. So, in our final model, we will be trying logcosh, mean absolute percentage error, and mean squared error (to see our improvement) and their performances.-

Table 8: Loss Function Selection

Errors and biases for different models with batch sizes 16 and 32										
	batch size=16					batch size=32				
	MAE	Age bias	Gender bias	Ethnicity bias	Face Expression bias	MAE	Age bias	Gender bias	Ethnicity bias	Face Expression bias
MSE	11,6451	8,9922	0,3722	1,2095	1,4066	13,1430	9,3686	1,6029	0,9615	1,3695
Sparse Caterogical Crossentrophy	12,9867	8,5758	0,3646	2,7435	1,6055	12,2119	9,4774	0,3514	1,0855	1,9479
Poisson	15,2839	6,5351	1,8649	1,8721	1,2170	11,7941	7,2527	0,4036	1,2445	1,9208
Huber	16,2971	12,4130	0,6091	1,8972	1,7770	12,5541	9,0745	0,0401	1,0695	1,4815
LogCosh	10,7871	11,4531	0,2327	0,5088	2,2603	12,6083	9,4870	1,0489	1,1571	1,5113
Mean Squared Logaritmic Error	12,8667	8,5768	0,3546	1,7435	1,4055	11,4020	7,2750	0,4406	0,8398	0,7045
MeanAbsolutePercentageError	**9,5153**	12,2073	1,1541	1,3389	1,6512	14,0351	7,0835	1,7299	0,6470	1,2351

e. Adding Weights

In the final step, we added weights to our loss function and have a look at how it improves the performance. We created 4 different sets of weights: age, facial expression, ethnicity, and gender. When we check the sizes of subgroups in our dataset we see that there are some big disproportions within groups:

- Age: 796 people < 20 , 2389 people from 20 to 40, 732 people from 40 to 60, and 148 people older than 60.
- Gender: 2068 males and 1997 females.
- Face expression: 1404 neutral face expression, 712 happy, 1784 slightly happy, and 165 with other facial expressions.
- Ethnicity: 3522 Caucasians, 119 Afro-Americans, and 424 Asians.

That's why we decided to check the results for the best models from the previous part (models with LogCosh, MeanAbsolutePercentageError, and MSE loss functions) with different weights and the results were as below in table 9.

Results are relatively hard to interpret as different models did well for different purposes. However, as mean absolute percentage error with gender weights did the best on both mean absolute error (our priority) and face expression bias and it did not do very bad on gender and ethnicity bias (although its performance on age bias is very bad), we decided to finalize our model with it.

Table 9: Weights Selection

Errors and biases for different models with weighted samples										
	gender weights					age weights				
	MAE	Age bias	Gender bias	Ethnicity bias	Face Expression bias	MAE	Age bias	Gender bias	Ethnicity bias	Face Expression bias
MSE	9,3128	**1,0803**	0,2238	1,4548	0,4217	**8,8081**	**1,9140**	**0,1422**	**0,8456**	0,5426
LogCosh	7,6004	1,7211	**0,0798**	**0,3802**	0,5424	9,0690	2,4891	0,5032	1,0872	**0,4626**
MeanAbsolutePercentageError	**7,0228**	4,3971	0,1855	0,9509	**0,1992**	10,0320	1,3406	0,3122	1,4495	1,2061
	face expression weights					ethnicity weights				
	MAE	Age bias	Gender bias	Ethnicity bias	Face Expression bias	MAE	Age bias	Gender bias	Ethnicity bias	Face Expression bias
MSE	**8,2783**	**1,0268**	0,2012	**1,0437**	**0,6654**	9,2434	3,3517	0,4660	0,6186	**0,5853**
LogCosh	8,4616	1,7380	**0,0099**	1,5095	0,9419	**7,2234**	**1,1942**	**0,1531**	0,7430	0,6018
MeanAbsolutePercentageError	23,0039	7,1069	3,2046	2,4049	2,2875	7,3863	1,7325	0,1649	**0,1408**	0,7067

Our final model is a ResNet101 model which is modified with 3 FC layers. Initialized with ImageNet weights and at first, only the newly added layers were trained, then the full model is trained. Hyperparameters are tuned as 32 batch size, Adam optimizer, 1e-5 learning rate. Our loss function is the mean absolute percentage error with weights based on gender. With this model, our final performance is summarized as follows:

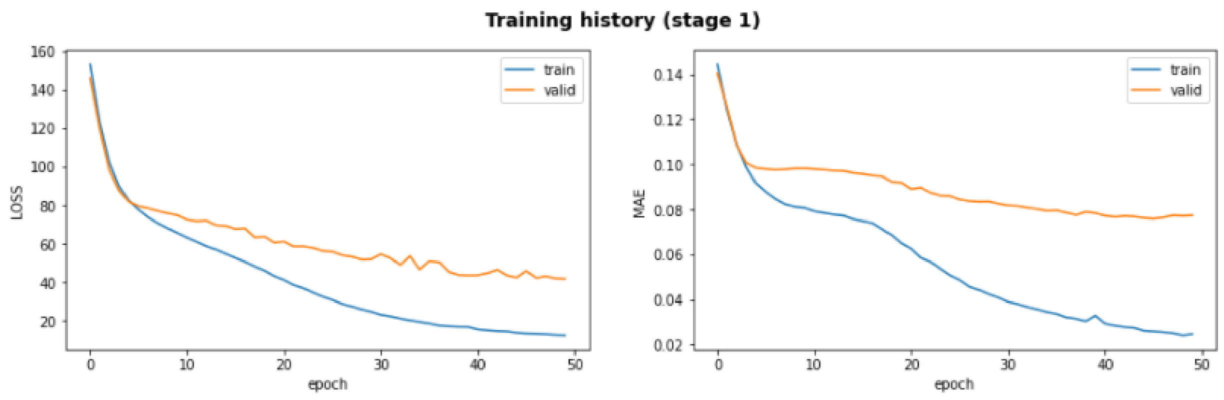

Figure 2: Final Validation Error

Table 10: Accuracy and Bias scores of our Final Model

Model	***Learning rate***	***Training strategy***	***Gender bias***	***Expression bias***	***Ethnicity bias***	***Age bias***	***MAE***
ResNet101 (after Loss Function. Optimization)	*1e-5*	*2 (see part 2-b)*	0.1855	0.1982	0.9509	4.3971	7.0228

We managed to improve all of our matrices.

3. FINAL REMARKS

In this assignment, we tried to build a model for age regression. First, we selected our backbone. We tried different versions of ResNet and ResNet101 came out to be superior in terms of validation loss/ execution time performance. We used ImageNet weights as initialization and first, we trained only the added layers for 25 epochs, then we trained the whole model with another 25 epochs. After fine-tuning hyperparameters, we decided on a loss function & weights based on validation loss and bias scores. Here, the mean absolute percentage error with weights on gender came out to be superior. Table 11 below shows that, when we compare our model with the data augmentation model that we previously did or with the model that is before the loss optimization, our loss function optimized model is superior in 4 out of 5 metrics.

Table 11: Comparison with Data Augmentation

Model	***Learning rate***	***Training strategy***	***Gender bias***	***Expression bias***	***Ethnicity bias***	***Age bias***	***MAE***
ResNet101 (before Loss)	*1e-5*	*2(see part 2-b)*	0.949	1.664	1.546	10.17	10.106
ResNet101 (after Loss)	*1e-5*	*2(see part 2-b)*	0.1855	0.1982	0.9509	4.3971	7.022
ResMet101 (data Aug.)	*1e-5*	*2(see part 2-b)*	0.235	0.658	0.778	6.300	7.241