CM502650: Machine Learning 2

Assignment 1: Age Estimation and Gender Classification

March 21 2022

Group number: 40

Student Number	Candidate Number	Contribution
159055506	08257	100%
219491783	08240	100%
219106485	08101	100%

Google Drive Links for models:

ModelA:
https://drive.google.com/file/d/12aLJteYCVf6ompfHQp5_K2mbUl0C_HkW/view?usp=sharing

ModelB:

https://drive.google.com/file/d/1-08z-KKKOmp4fA5KvS1PD6rHHK7Yf7OD/view?usp=sharing

Introduction

In the past years there has been substantial interest and research into the topic of age and gender estimation using face images. In this assignment, we construct and train two CNN (Convolutional Neural Network) models. The model's outputs age and gender estimates for a given 128x128 image of a face. The dataset contains 5,000 labeled face images that was split for training and validation. This dataset is a subset of the popular UTKFace dataset consisting of over 20,000 face images.

For the first model, we define our own CNN from scratch. Different techniques are considered for building the model architecture to tackle common problems in neural network training such as vanishing gradient problem, underfitting, overfitting, and the small dataset problem.

For the second model, we built a CNN network based on the InceptionV3 pre-trained ImageNet model. After careful research and comparison to different pretrained models the inceptionV3 was chosen as it produced the best results due to its very deep architecture and training on 1.2 million images.

Age estimation and gender prediction models are still being improved due to the impactful and many applications they have. As a result of testing out different architectures and techniques to handle the problems addressed above, we gain a better understanding of the practical considerations involved with CNNs.

Dataset and Augmentation

The images provided are labelled with information on age, race, and gender but for the purposes of this assessment we only investigate the age and gender. The age is an integer ranging between 0 to 116 while the gender only represents female and male with 0 and 1 respectively. The data was parsed and stored in a data frame for later pre-processing. The data augmentation was done using the data generator where adjustments were made to the data such as rotations, illuminations, and rescaling.

Our own CNN

Before building our own CNN we researched best practices through examples online, reading papers, and module material (Sharma, N., 2022, Bressan 2020).

Our decision for the architecture was starting with a single input for the image we are feeding the CNN then decomposing into 2 branches composed of different layers followed by their respective Dense layers. The default structure for our model is based on repetition of the following layer structure: a Conv2D layer with a Relu activation layer, followed by a BatchNormalization layer, a MaxPooling, and then finally a Dropout layer As in all CNN architectures, several convolution layers are used, with a doubling in the number of filters (16->32->64->128) as proceeding deeper into the architecture. Convolution layers characterize CNNs and differentiate them from standard Multi-Layer Perceptrons; they allow our model to learn ‘features’ in images.

Following every convolution layer is a ReLU activation layer. As detailed in the lecture notes, a ‘ReLU’ is a pixel-by-pixel operation that simply replaces negative values with zero. Such layers are crucial in introducing complexity and specifically non-linearity into artificial neural networks, including CNNs.

In our model, after the ReLU activation layers, we have ‘batch normalization layers. As we have learned in lectures, this helps avoid the vanishing gradient problem by attempting to achieve a ‘stable distribution of activation values throughout training’ (Loffe, 2015). Whilst it is suggested that these layers should be inserted before the activation layers, in practice for our dataset and problem, we found that our model performed better with these layers after the activation layers.

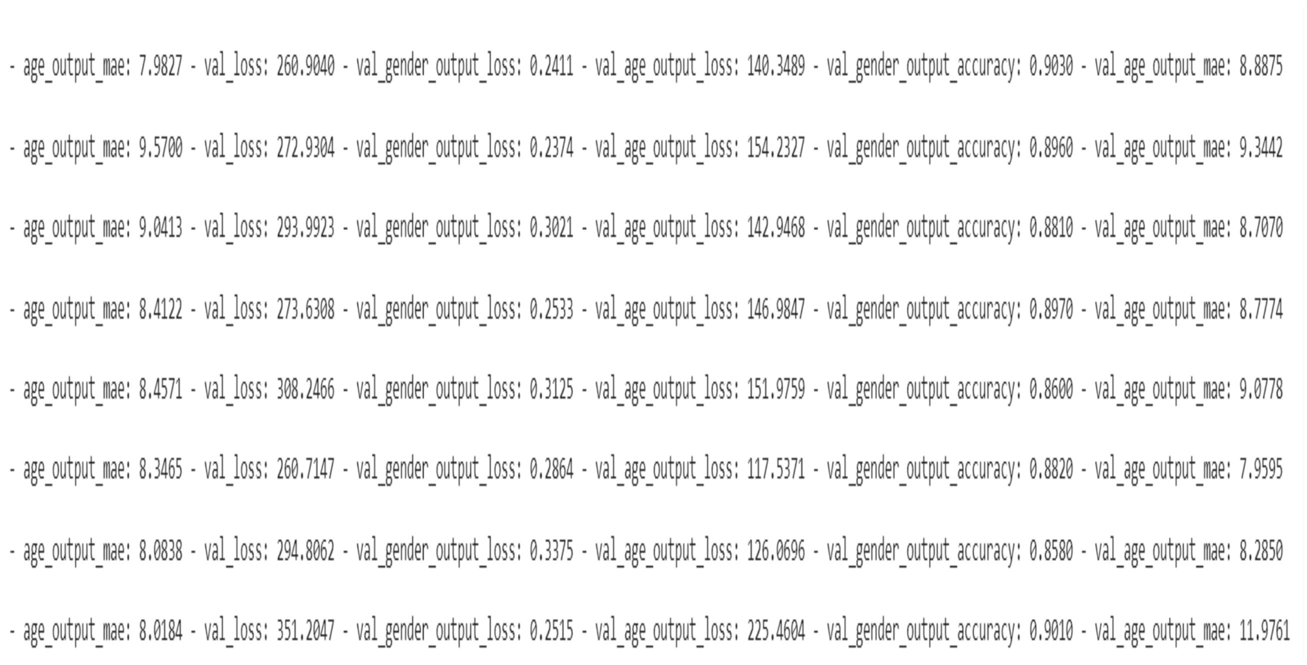
After this, the standard pooling layers are used. These layers are necessary for CNNs; they reduce the dimensions of the feature maps and stop the number of parameters to be learned in training from increasing exponentially due to the convolution layers. Average pooling was considered and tested, but as other practitioners have found, max-pooling proved preferable.

For the last part of the model, the layer outputs are flattened. These flattened outputs are fed into fully connected (dense) layers, followed by some more ReLU activation layers. Finally, the final output layers are 1-unit; with a sigmoid activation for the gender branch (to output a probability between 0 and 1), and a linear activation for the gender branch (as predicting age is a regression problem, and we have not normalized age).

Several dropout layers are inserted throughout the network. These reduce overfitting by ignoring a random selection of neurons during training. By this, it is meant that these neurons are not considered during a particular forward or backward pass.

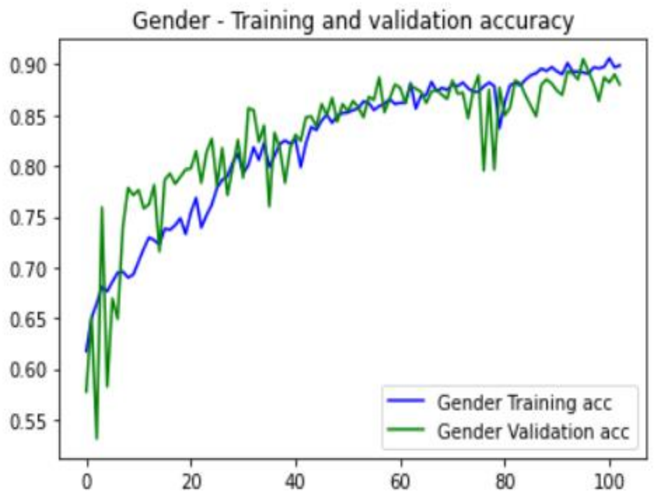
Hyperparameters such as augment parameters (e.g. max rotation), batch size, learning rate, dropout %, and pool size were selected by a combination of manual testing and research into common practice online. Using Keras Tuner was not as effective as manually experimenting with the hyperparameters and other hyperparameter techniques were tricky to implement due to the structure of our model.

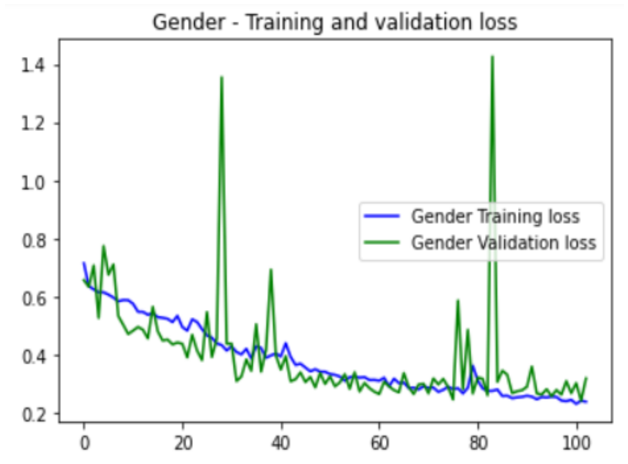
Screenshot of training outputs:



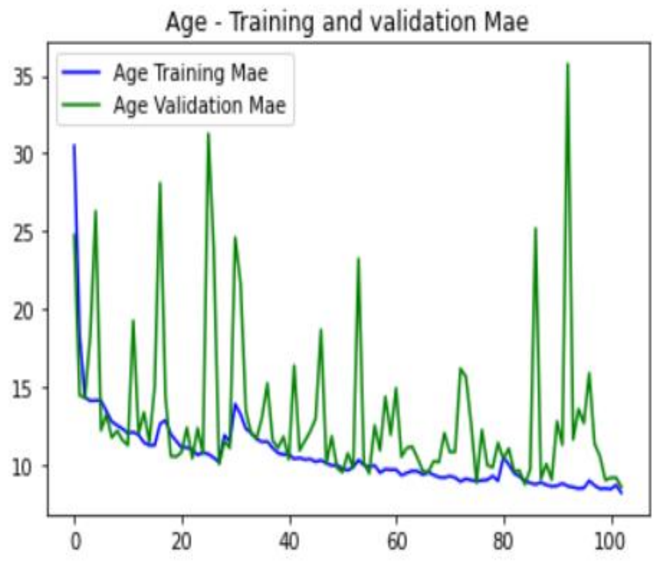
```
- age_output_mae: 7.9827 - val_loss: 260.9040 - val_gender_output_loss: 0.2411 - val_age_output_loss: 140.3489 - val_gender_output_accuracy: 0.9030 - val_age_output_mae: 8.8875

- age_output_mae: 9.5700 - val_loss: 272.9304 - val_gender_output_loss: 0.2374 - val_age_output_loss: 154.2327 - val_gender_output_accuracy: 0.8960 - val_age_output_mae: 9.3442

- age_output_mae: 9.0413 - val_loss: 293.9923 - val_gender_output_loss: 0.3021 - val_age_output_loss: 142.9468 - val_gender_output_accuracy: 0.8810 - val_age_output_mae: 8.7070

- age_output_mae: 8.4122 - val_loss: 273.6308 - val_gender_output_loss: 0.2533 - val_age_output_loss: 146.9847 - val_gender_output_accuracy: 0.8970 - val_age_output_mae: 8.7774

- age_output_mae: 8.4571 - val_loss: 308.2466 - val_gender_output_loss: 0.3125 - val_age_output_loss: 151.9759 - val_gender_output_accuracy: 0.8600 - val_age_output_mae: 9.0778

- age_output_mae: 8.3465 - val_loss: 260.7147 - val_gender_output_loss: 0.2864 - val_age_output_loss: 117.5371 - val_gender_output_accuracy: 0.8820 - val_age_output_mae: 7.9595

- age_output_mae: 8.0838 - val_loss: 294.8062 - val_gender_output_loss: 0.3375 - val_age_output_loss: 126.0696 - val_gender_output_accuracy: 0.8580 - val_age_output_mae: 8.2850

- age_output_mae: 8.0184 - val_loss: 351.2047 - val_gender_output_loss: 0.2515 - val_age_output_loss: 225.4604 - val_gender_output_accuracy: 0.9010 - val_age_output_mae: 11.9761
```

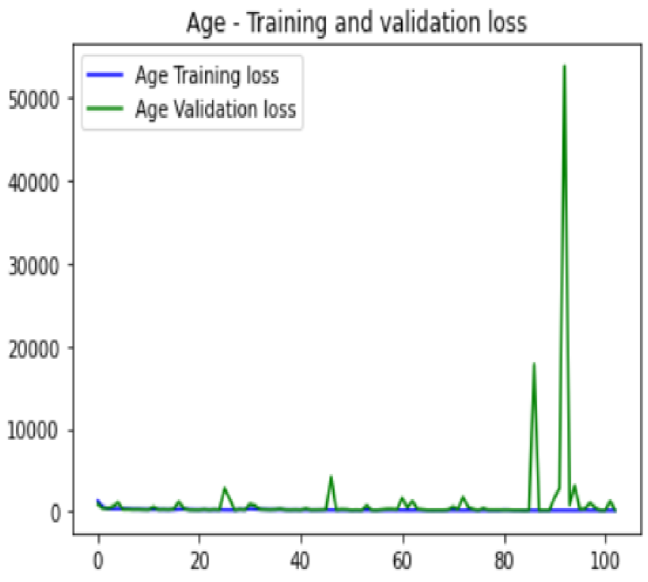
See below the training and validation curves for the loss, accuracy (for gender) and mean absolute error (for age); for modelA (our CNN):





The gender training and validation accuracies are performing well on the data and increase to stabilize close to 90% accuracy where the gap between the curves is very small indicating a good model. The loss for the gender seems to be exploding at times which can be contributed to our small validation set but is generally decreasing and stabilizes near 0.2 thus also indicating a good fit for the model.





The age MAE on the validation set reaches a minimum of 6.9 where the validation and training gap was minimal but there was a fluctuation in the validation set that could be contributed to dataset size. Moreover, the validation loss seems to decrease and stabilizes near-zero except for a few fluctuations.

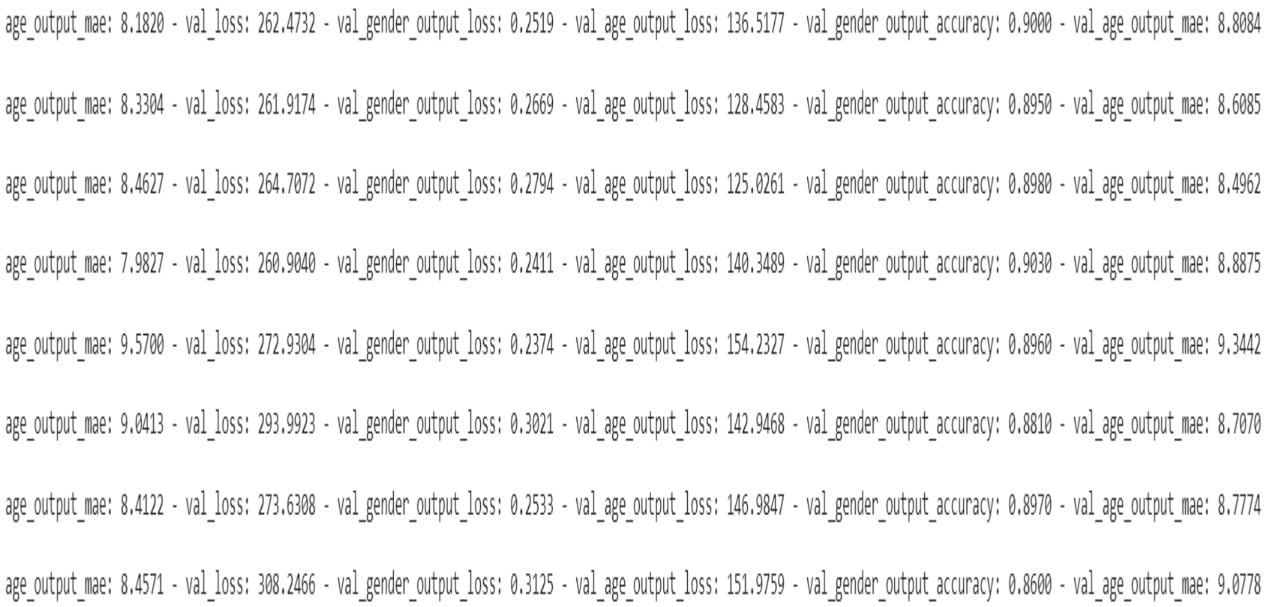
Pre-trained CNN

The 3 models considered for our pre-trained model are well known for their high performance: VGG16, InceptionV3, and VGGFace2. Although the VGGFace2 model would have produced the best outputs, we faced implementation problems to our task. Meanwhile, the VGG16 produced good models but lacked complexity to fit the data as observed in the fluctuations in training. Therefore, InceptionV3 was chosen in response to VGG16 performance due to its higher complexity and greater depth.

For our pre-trained CNN, InceptionV3 was used as the base model. The model has achieved over 90% top-5 accuracy on the ImageNet test dataset containing 1000 image classes (Koehrsen,2017). The architecture consists of over 59 convolution layers with decreasing filter size and incremental increases in the number of filters being assessed. However, the InceptionV3 model is limited to identifying the 1000 different images of which it was trained. To classify different classes or in our case perform gender classification and age estimation tasks, we would need to train the parameters for the final layers of the model. The lower layers of the CNN are already very good at identifying lower-level features such as shapes and colours therefore it is enough to train the final layers of the model. Moreover, we would need to modify the output of the CNN and customize our own as we require a multi-output model for our tasks.

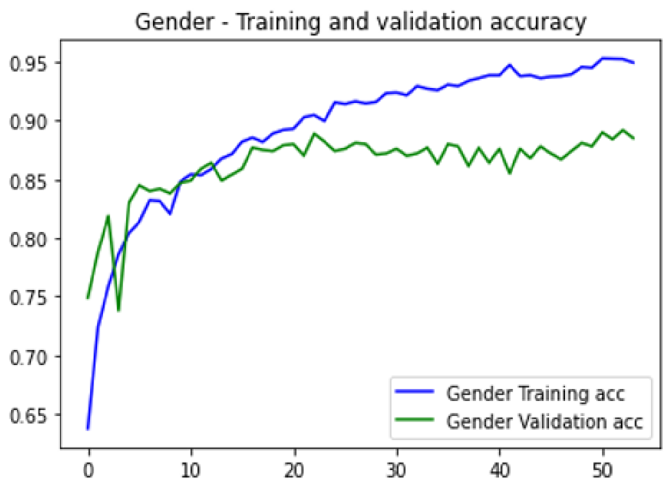
Similar to the previous CNN architecture, the two branches were split off immediately which allowed the CNN to learn features and parameters specific to the two problems it is tasked with. Activation layers (ReLUs) were used to introduce more complexity and non-linearity. Dropout layers and batch normalization layers are used. Similarly, to above, they help against overfitting and the vanishing gradient problem. At first, the model performed well but suffered from overfitting and so Dropout and Gaussian noise layers were added sequentially. Also, early stopping was used to prevent overfitting.

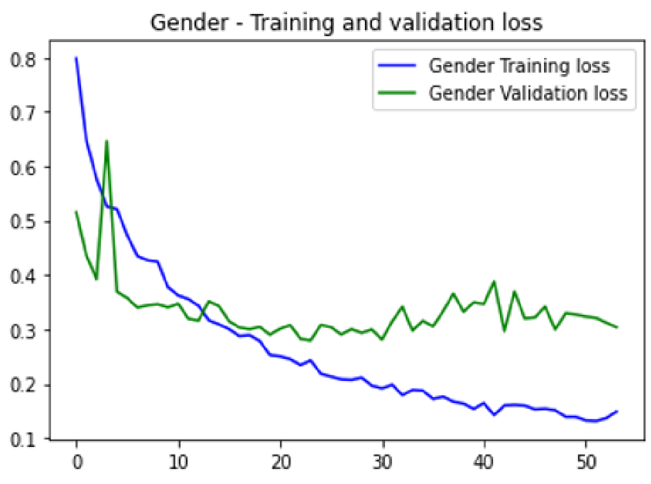
Screenshot of training outputs:



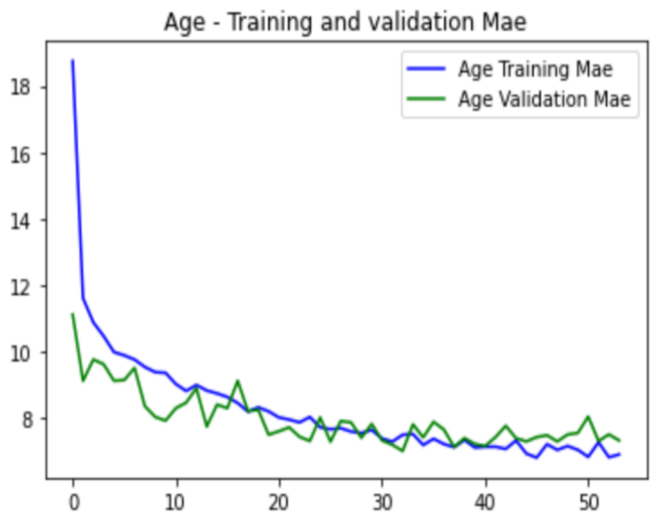
```
age_output_mae: 8.1820 - val_loss: 262.4732 - val_gender_output_loss: 0.2519 - val_age_output_loss: 136.5177 - val_gender_output_accuracy: 0.9000 - val_age_output_mae: 8.8084

age_output_mae: 8.3304 - val_loss: 261.9174 - val_gender_output_loss: 0.2669 - val_age_output_loss: 128.4583 - val_gender_output_accuracy: 0.8950 - val_age_output_mae: 8.6085

age_output_mae: 8.4627 - val_loss: 264.7072 - val_gender_output_loss: 0.2794 - val_age_output_loss: 125.0261 - val_gender_output_accuracy: 0.8980 - val_age_output_mae: 8.4962

age_output_mae: 7.9827 - val_loss: 260.9040 - val_gender_output_loss: 0.2411 - val_age_output_loss: 140.3489 - val_gender_output_accuracy: 0.9030 - val_age_output_mae: 8.8875

age_output_mae: 9.5700 - val_loss: 272.9304 - val_gender_output_loss: 0.2374 - val_age_output_loss: 154.2327 - val_gender_output_accuracy: 0.8960 - val_age_output_mae: 9.3442

age_output_mae: 9.0413 - val_loss: 293.9923 - val_gender_output_loss: 0.3021 - val_age_output_loss: 142.9468 - val_gender_output_accuracy: 0.8810 - val_age_output_mae: 8.7070

age_output_mae: 8.4122 - val_loss: 273.6308 - val_gender_output_loss: 0.2533 - val_age_output_loss: 146.9847 - val_gender_output_accuracy: 0.8970 - val_age_output_mae: 8.7774

age_output_mae: 8.4571 - val_loss: 308.2466 - val_gender_output_loss: 0.3125 - val_age_output_loss: 151.9759 - val_gender_output_accuracy: 0.8600 - val_age_output_mae: 9.0778
```

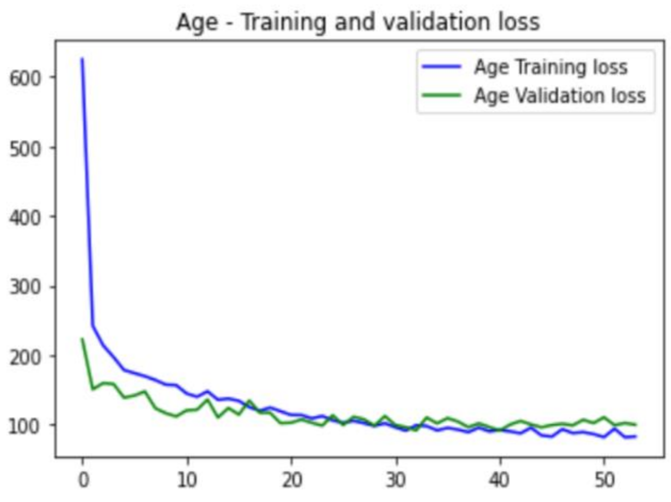
See below the training and validation curves for the loss, accuracy (for gender) and mean absolute error (for age); for modelB (pre-trained CNN):





Both the training and validation are increasing with training being higher than validation, but the graph shows a little overfitting due to the gap between the curves. We set our model to train for 100 epochs and batch size of 50 but due to early stopping it stopped early. The train and test loss are both decreasing as epochs increase and stabilize close to zero towards the end. This demonstrates a good fit learning curve for the model.





Similarly, the train and test loss for age estimation is decreasing and stabilizes close to 100 towards the end thus demonstrating a good fit learning curve for the model.

Summary and Discussion

As can be seen in the plots, the pre-trained model outperforms our model built from scratch. The CNN build from scratch contains a lot of fluctuation and reaches an accuracy close to 90%. Even though the pre-trained model doesn't outperform greatly (only by 2%), the plots show much less fluctuations and that the model is a better fit through the loss and accuracy curves. This is expected, since the pre-trained model has seen millions more images from which to learn features, has spent much more time learning such features, and has a much more complex architecture.

Whilst image augmentation can allow us to increase the number of images we train on for our model, 5000 images (4000 for training) is still a relatively small amount, particularly given the fact that we are trying to train a single model with two outputs. The amount of data holds a limitation on our model's performance and with more data it would produce more accurate results.

It is difficult to see briefly when looking at the learning curves, but one thing we could observe during training was that an improvement in performance (in loss or accuracy/MAE terms) in one output would often correspond to a drop in performance for the other output. The model seems to struggle to train its parameters to suit *both* problems.

An attempt to rectify this issue was made, by splitting the branches 'earlier' on in the depth of the network. Hence, it was hoped the model would essentially learn two independent networks: with marginally.

To conclude, this assessment taught us the process of implementing a CNN and how to interpret and improve such models using various techniques. Whilst we managed to construct and train models we are satisfied with; it will be interesting to discuss with other groups how they solved the issues we faced.

References

Bressan, R. (2020, June 2). Building a multi-output convolutional neural network with Keras. Medium. Retrieved March 21, 2022, from https://towardsdatascience.com/building-a-multi-output-convolutional-neural-network-with-keras-ed24c7bc1178

Koehrsen, W. (2017, August 20). *Facial recognition using Google's Convolutional Neural Network*. Medium. Retrieved March 21, 2022, from https://williamkoehrsen.medium.com/facial-recognition-using-googles-convolutional-neural-network-5aa752b4240e

Sharma, N., Sharma, R. & Jindal, N. Face-Based Age and Gender Estimation Using Improved Convolutional Neural Network Approach. Wireless Pers Commun (2022). https://doi.org/10.1007/s11277-022-09501-8