Age Estimation and Gender Classification Report

1. INTRODUCTION

The goal of this project is to construct two CNN models, one trained from scratch and the other by fine-tuning a pre-trained model, capable of predicting a person's age and gender from face images. Each model is trained and validated on 5000 labelled face images of size 128x128, taken from the UTKFace dataset. Each model is to then be tested on unseen data, measuring the performance through the gender classification accuracy and age estimation MAE (mean absolute error).

2. The custom CNN

The custom CNN consists of an input layer, a data augmentation layer and three blocks of convolution, batch normalisation, ReLU activation function and maxpooling. The CNN then splits off into two separate branches for age and gender: each branch contains two of the aforementioned blocks followed by a flatten layer, a dense layer, batch normalisation, ReLU activation function, dropout and a one neuron dense layer. A linear activation function was used for age estimation and a sigmoid activation function was used for gender classification to give a value between 0 and 1. (See Figure 1).

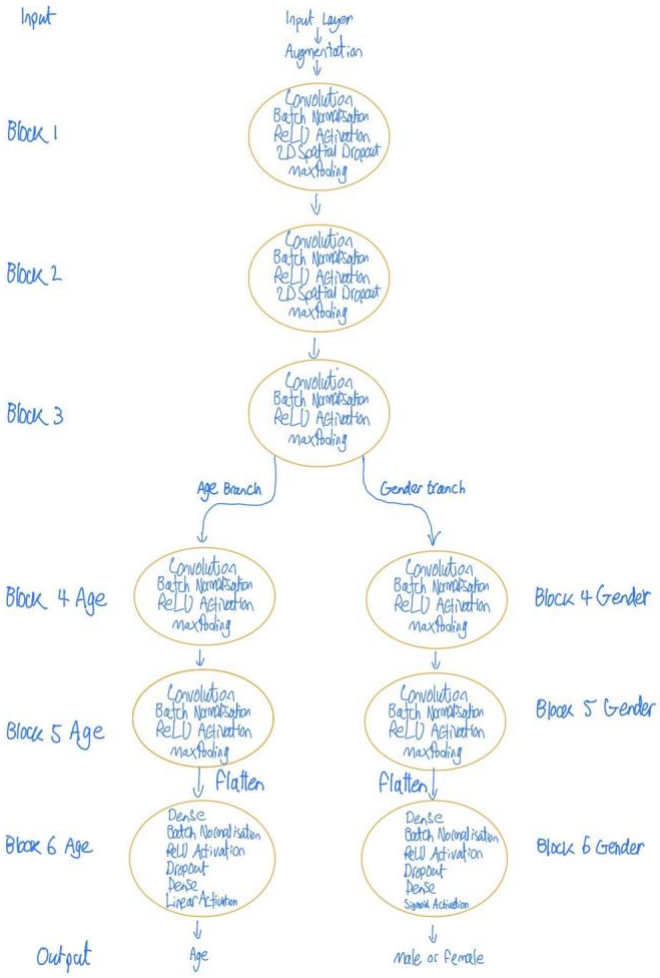

Figure 1. Model A Architecture

Batch normalisation was used to "achieve a stable distribution of activation values throughout training" (Ioffe and Szegedy, 2015) and was applied before nonlinearity after experimentation for best performance;

application before ReLU activation prevents huge outputs. Batch normalisation acts as a regulariser and was used in addition to dropout to prevent overfitting; dropout was applied after the penultimate dense layer to prevent over reliance on any of these neurons. Spatial dropout drops entire 2D feature maps instead of individual pixels and was used in place of regular dropout after convolutional layers. This has been shown by Lee and Lee (2020) to be more effective than element-wise dropout in preventing "co-adaptation between neurons on the same channel of a convolutional layer". Weight initialisation was also used (He uniform with ReLU and Glorot normal with sigmoid) to prevent vanishing gradients. An adaptive learning rate of Adam optimiser, in addition to a learning rate scheduler (`'keras.callbacks.ReduceLROnPlateau'),` which decreases the learning rate by a factor of 0.8 if validation loss stops improving after 7 epochs, were used for better performance. Data augmentation was used to address the issue of only 5000 training samples by generating new synthetic data through flipping, rotating, zooming in on, translating and shearing the images. Binary cross entropy loss was used for gender output and mean squared error (MSE) loss was used for age output because it penalises large errors at a greater rate than MAE (Richmond Alake, 2023).

The training process started with adjusting the model hyperparameters (the topology and size of the CNN), before adjusting algorithm hyperparameters (learning rate etc.). My goal was to maximise training performance and then add regularisation techniques to maximise validation performance, as described by Karpathy (n.d.). I started by optimising the number of identical convolution, batch normalisation, ReLU activation and maxpooling blocks and adjusting where to split the model from one branch into two. Since many of the generic features being extracted would be useful for both identifying gender and predicting age, both branches share earlier layers. Additionally, I adjusted the number of fully connected layers at the end of each branch. I then experimented with increasing the number of neurons, the number of filters (and sizes) and the number of epochs to stop underfitting. I then added overfitting measures (dropout and spatial dropout), shifting my focus to validation performance. A learning rate scheduler was added for

performance and the initial learning rate was increased from a starting value of 0.001 to 0.0013 to allow faster convergence.

The learning curves (see Figure 2) indicate no signs of overfitting as gender and age validation loss does not increase majorly at any point and closely follows the training loss. The loss (in validation and training) also decreases at a fast rate which suggests there is no underfitting either. Similarly, the gender accuracy and age MAE in validation closely follows performance in training and both improve at a fast rate, indicating no overfitting or underfitting. The model actually performs better on validation in all four curves over the first 75 epochs. This is likely due to dropout and data augmentation not being applied in validation, so all neurons are active and the model is validated on unaltered face images, improving performance.

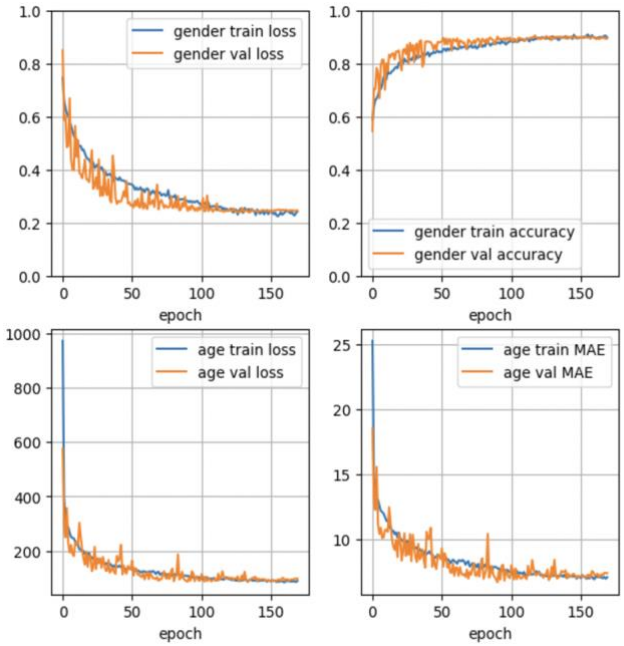

Figure 2. Model A Learning Curves

3. The pre-trained CNN

The pre-trained CNN consists of an input layer, a data augmentation layer, three image preprocessing layers, a VGG-16 model (excluding the top), followed by 2D global average pooling and dropout. The CNN then splits off into two separate branches for age and gender. The age branch consists of dropout, two blocks of a dense layer, batch normalisation, ReLU activation function and dropout and a final one neuron dense layer with linear activation function. The gender branch consists of two of the aforementioned blocks followed by a final one neuron dense layer using sigmoid activation function. (See Figure 3).

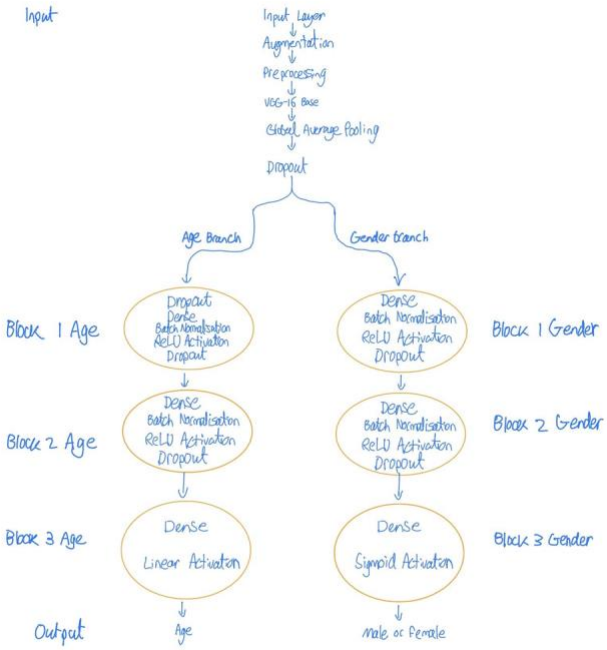

Figure 3. Model B Architecture

To fine-tune the VGG-16 model trained on ImageNet for age and gender prediction, gradual unfreezing, as described by Howard and Ruder (2018), was used to prevent "catastrophic forgetting" (learned features being forgotten due to unfreezing all layers at once). The VGG-16 base is frozen for 5 epochs and only the fully connected layers are allowed to train, followed by the last 8 layers (2 blocks) of the VGG-16 being unfrozen and able to learn. The earlier layers are kept frozen since features learnt by the earlier layers of VGG-16 (edges, corners etc.) are more generic and shared in this task. Freezing earlier layers allows just the later layers to learn features specific to age and gender. For the first 5 epochs, a learning rate of 0.001 (with RMSProp) was used and then a smaller learning rate of 0.00002 (with AdamW) was used after unfreezing so as to not unlearn the initial weight initialisations (Thor, 2016). AdamW was used since it generalises better than Adam, allowing similar performance on image classification to SGD with momentum (Loshchilov and Hutter, 2019).

Global average pooling was used over flatten in order to minimise the number of parameters being learnt (Yamashita et al., 2018). Data augmentation, batch normalisation (before ReLU activation), weight initialisations, learning rate scheduling, dropout and MSE loss were all used as in the custom CNN, adjusting parameters for best validation performance. Loss weights of 1.0 and 10.0 for age and gender were used to focus on minimising the gender loss (since the model was struggling to predict gender):

$$L_{\text{total}} = 1.0 \cdot L_{\text{age}} + 10.0 \cdot L_{\text{gender}}$$

The learning curves (see Figure 4) for gender show that the model has better performance (higher accuracy and lower loss) on validation than training due to data augmentation and dropout being removed. Since the model is generalising this well, it is not overfitting to the training data. The model's age validation loss and MAE do not indicate major overfitting (validation and training are very similar), but have plateaued and stopped improving. The age loss falls rapidly and the gender loss falls to a very low level which do not imply underfitting; when adding more complexity (more neurons, layers etc.) the model begins to overfit so the model is not underfitting.

Tensorflow.keras.callbacks.ModelCheckpoint was used in both models to reload the model with the best validation performance before any possible overfitting takes place.

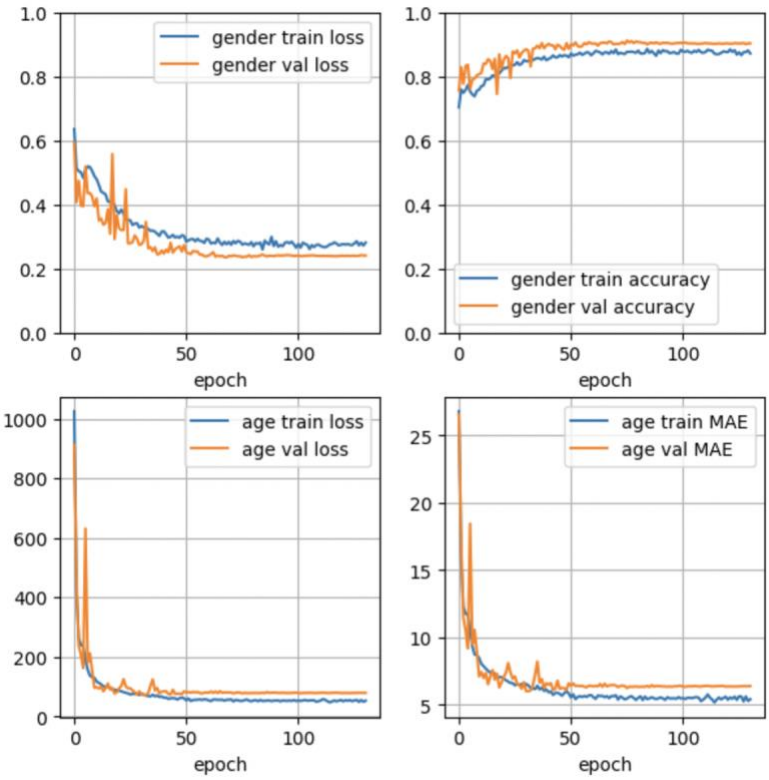

Figure 4. Model B Learning Curves

4. SUMMARY AND DISCUSSION

The pre-trained CNN outperforms the custom CNN when comparing the results of the 'best model' (the model saved by ModelCheckpoint) on validation data. The pre-trained CNN achieved 91.2% gender accuracy and age MAE of 6.25 compared to 90.0% gender accuracy and age MAE of 6.58 by the custom CNN. The learning curves also indicate there is less variance from epoch to epoch in the pre-trained model (less jumps and shocks in the curves), implying more stable and reliable performance in testing. The pre-trained CNN also requires less epochs for training, though this is offset by having more trainable parameters (1449474 vs 1015906) than the custom CNN so both models take about the same amount of time to train.

The models subjectively perform better on age than gender when compared to a person; people are generally better at identifying gender than predicting someone's age using a face image and would likely outperform both models' gender accuracy, but less likely the age MAE. Both models are very strong in age estimation, comparable to a person. Improvements to gender classification could be made through using a model pre-trained on a more similar task or having one model devoted to gender classification and another for age estimation: sometimes there was a trade off between getting high gender accuracy and getting low age MAE (demonstrated by the use of loss weights).

The limiting factor of both models appeared to be the 5000 image samples as no more added complexity was able to improve performance: this amounts to just 21 face images per gender and age combination, on average. More data (if possible) or additional data augmentation techniques (blurring, added noise, contrast) could be considered to further improve performance.

Carrying out this task, it became evident how transfer learning and techniques like data augmentation might be helpful in real world contexts where there is limited training data (such as medical scans of rare conditions): data augmentation to provide more images and transfer learning to improve and stabilise performance are vital. Additionally, it only seems feasible to construct a custom model when there is lots of training data available. Otherwise, transfer learning makes more sense.

Word Count: 1564

5. REFERENCES

Alake, R. (2023). *Loss Functions in Machine Learning Explained*. [online] Datacamp.com. Available at: https://www.datacamp.com/tutorial/loss-function-in-machine-learning.

Howard, J. and Ruder, S. (n.d.). *Universal Language Model Fine-tuning for Text Classification*. [online] Available at: https://arxiv.org/pdf/1801.06146 [Accessed 9 May 2024].

Ioffe, S. (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. [online] Available at: https://arxiv.org/pdf/1502.03167.

karpathy.github.io. (n.d.). *A Recipe for Training Neural Networks*. [online] Available at: https://karpathy.github.io/2019/04/25/recipe/.

Lee, S. and Lee, C. (2020). Revisiting spatial dropout for regularizing convolutional neural networks. *Multimedia Tools and Applications*. doi:https://doi.org/10.1007/s11042-020-09054-7.

Loshchilov, I. and Hutter, F. (n.d.). *DECOUPLED WEIGHT DECAY REGULARIZATION*. [online] Available at: https://arxiv.org/pdf/1711.05101.

Thor, W.-M. (2016). *Fine-tuning vs. Feature Extraction: Advanced Considerations*. [online] Apxml.com. Available at: https://apxml.com/courses/cnns-for-computer-vision/chapter-6-advanced-transfer-learning-domain-adaptation/fine-tuning-feature-extraction-advanced [Accessed 27 Nov. 2025].

Yamashita, R., Nishio, M., Do, R.K.G. and Togashi, K. (2018). Convolutional Neural networks: an Overview and Application in Radiology. *Insights into Imaging*, [online] 9(4), pp.611–629. doi:https://doi.org/10.1007/s13244-018-0639-9.