Machine Learning for Imaging – Coursework Report
Age Regression from Brain MRI

Group: **53**

Tom Gotsman, Nikolas Theodosiou, Thomas Falezan
{tg220, nt220, tf317}@ic.ac.uk

1 Part A

Image Segmentation architecture, hyper parameters and results
For the image segmentation, we implemented a variant of a U-Net. The first part of the network consists of an encoder: the input image size is reduced multiple times and the number of feature maps increased to 64 (3d convolutions and max pooling layers used). The inputs are then passed in a bottle neck (image size and number of feature maps remain constant) and finally to a decoder: using transposed convolution and 3d convolution layers, we upsample the images and reduce the number of feature maps to 4 (number of classes). The intermediate outputs of the encoder are also fed to the decoder at intermediate stages.
To train the U-Net, we used image sizes of (96,96,96), the loss function used was the categorical cross entropy, the learning rate set to 0.001 and the batch size to 6. This combination of parameters seem to yield the best results. Both the dice scores and the loss were recorded for training and validation set.

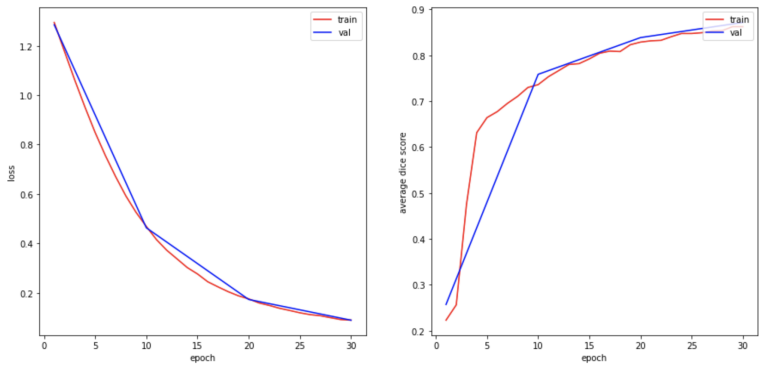


Figure 1: Segmentation loss curves and average dice scores

The network gives relatively good results: the average dice score reaches 0.86 compared to 0.7 with the original model. Figure 2 also shows that it performs best on the white matter while having more difficulty segmenting out the CSF tissue.

Feature calculations and plots
We created features to use in the age regression process by computing the relative volumes as the ratios between each tissue volume and overall brain volume. We attempted to create other features using polynomials, but as can clearly be seen in the notebook these did not give better results. Fig 6 shows that for GM and CSF tissues we can see a correlation between the volume and the age, however for WM this correlation does not appear to exist.

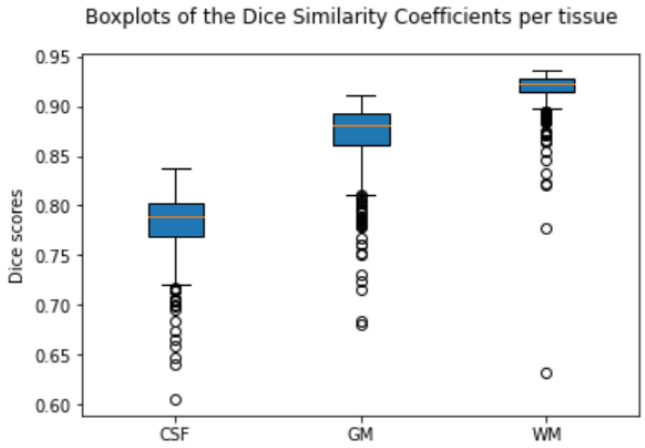


Figure 2: Boxplots of the dice scores per tissue

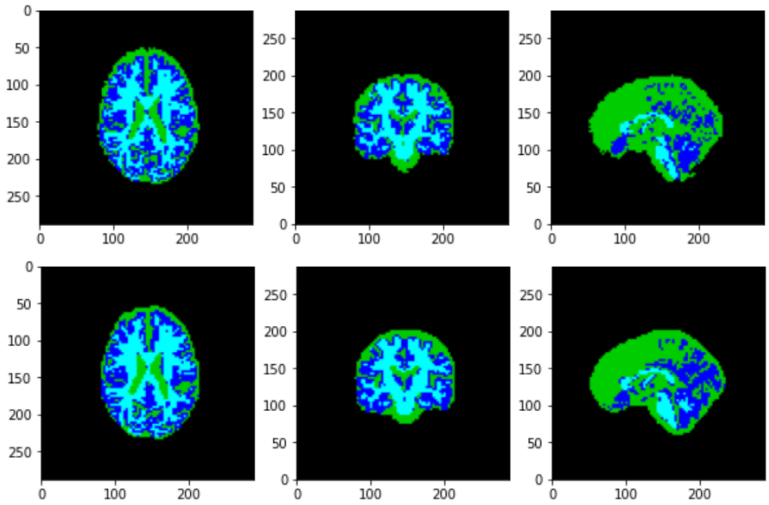


Figure 3: example of a reference (top) and predicted (bottom) segmentation

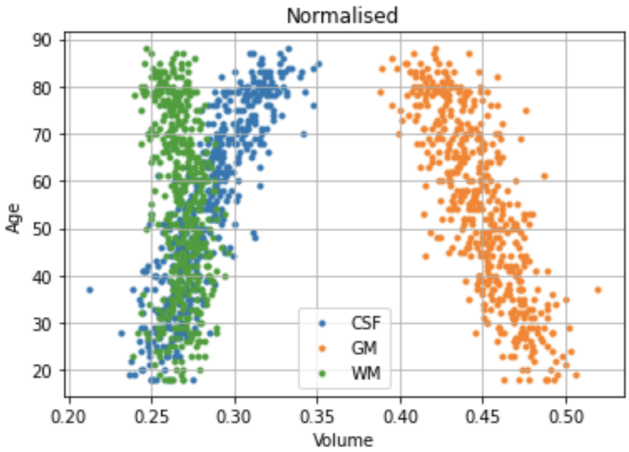
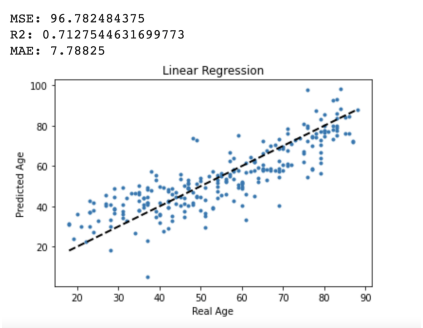


Figure 4: Normalised Tissue Volume vs Age.

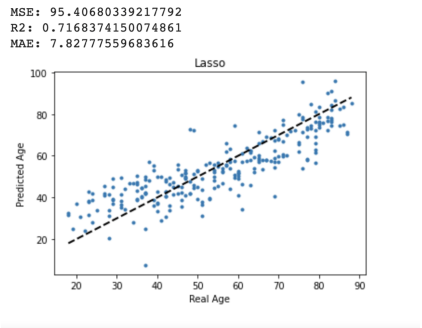
Regression models and cross validation results

We implemented three types of regression, linear, lasso and bayesian ridge. Linear regression does not penalise the model for its choice of weights. Lasso is a modification where the model is penalized for the sum of absolute values of the weights, therefore the absolute values of the weights are reduced, and many tend to zero. Bayesian ridge penalises the model for the sum of squared value of the weights,

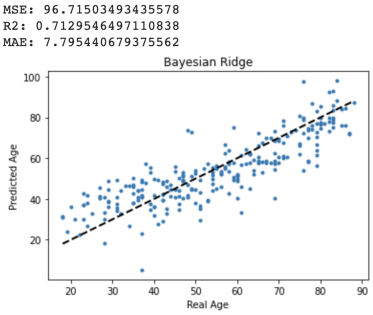
therefore penalizes the extreme weights.



Linear Regression MAE: 7.788



Lasso MAE: 7.828



Bayesian Ridge MAE: 7.795

2 Part B

As instructed, we started by improving the FCLeNet. As this is a regression task, we started by replacing the softmax activation with a linear activation. Further, we also replaced all 2d layers with corresponding 3d layers. Following this baseline, we tried several different implementations mostly using Conv3d+BatchNorm3d+ReLU Blocks. During the early stages of training, we experienced high bias, so we decided to increase the number of feature maps in our model. For downsampling our images, we tried both strided convolutions, and Maxpool3d, with the latter giving better results. To prevent the model from overfitting, we added a dropout layer before our fully connected layer in the end. Our final architecture is made up by 5 Conv3d(3x3x3)+ReLU()+BatchNorm3d()+MaxPool3d(2x2x2) blocks, followed by a dropout and a fully connected layer. Finally, we use a linear activation to perform the regression.

In terms of data pre-processing the images were preprocessed as in part A. Additionally, the patient ages were normalised before being fed to the model, for both the train and test set. For the 2 fold cross validation, we trained for 30 epochs on each fold, using a learning rate of 0.001. For the loss function, the MSE was used while the MAE was also recorded. The final results are shown below:

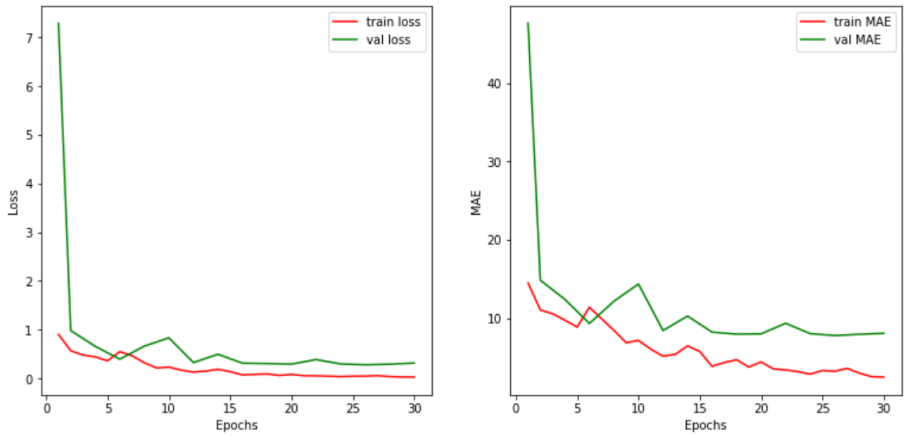


Figure 5: 2 Fold Cross Validations (Left MSE, Right MAE)

After getting satisfactory results (training and validation MAE under 10 and 5 respectively), we used larger input images (96x96x96). The final training was performed on 500 images, using 35 epochs and the same hyperparameters as before. Here, we used a batch size of 8. The 47 image dataset was used for validation with a batch size of 2. At epoch 35 the recorder validation and train MAEs were 5.66 and 2.44 respectively. Relevant plots follow below:

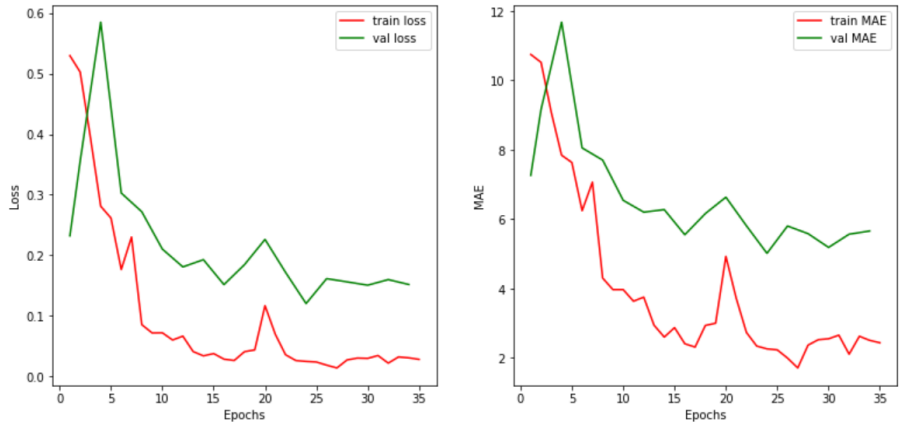


Figure 6: Training on all 500 Samples (Left MSE, Right MAE)

3 Age Regression Results

The two test set results are displayed in figure 7 (in part A, the linear regression model was kept as it seemed to give the best results). The plots look relatively similar, the points are scattered around the line y=x. It appears that the points in the second plot fit more closely the line but the difference is not clear. Indeed, the quantitative results suggest that the model in part B slightly outperforms part A's pipeline: the final MAE on the held out test set is 6.94 in part B compared to 7.43 in part A.

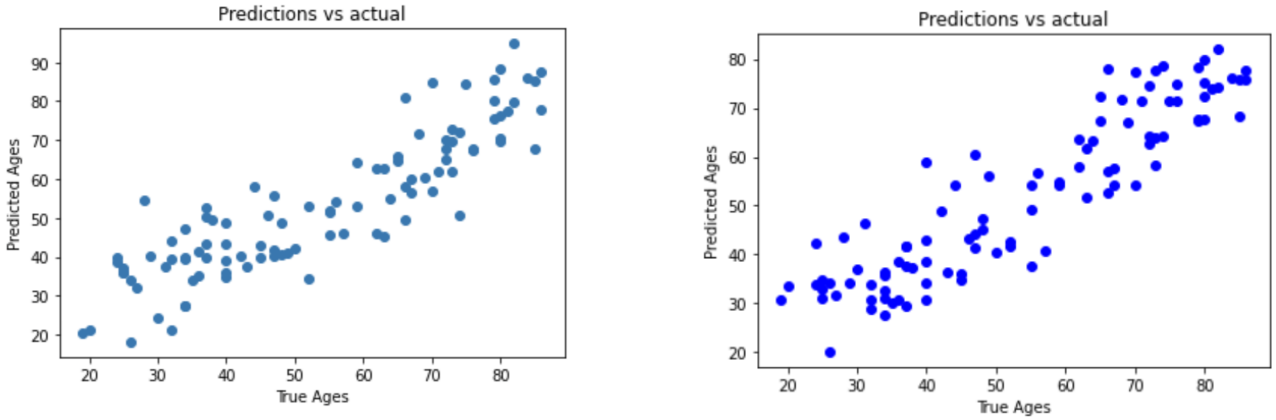


Figure 7: Scatter plots of the predicted ages versus true ages for Part A (left) and Part B (right)

The main difference between the two parts is that in part A, we split the problem into multiple subtasks (image segmentation, feature extraction and regression) whereas part B's pipeline is end-to-end: the age is predicted directly from the preprocessed scans. In part B, there is no need to handcraft the features, the model directly learns the ones most useful for the regression task. This explains why part B's model outperforms the entire part A pipeline.

Although part B's model provides better results, finding a good architecture and the right set of hyperparameters to reach a stable learning was more tedious. As the cross validation and final training plots suggest (figures 5 and 6), the training and validation losses exhibit relatively high variance and the entire procedure is quiet sensible to the choice of hyperparameters.

Finally, comparing the MAE obtained after cross validation to the MAE obtained on the held out test after training on the 500 samples, we notice two things. For part A, the two MAEs are relatively similar (7.8 and 7.4 respectively) although in the second case, the model is trained on twice the amount of data. This is most likely due to the fact that a linear regression model has a very small number of parameters. The model is not limited by the size of the dataset but its capacity. For part B, this is the

opposite: the model improves a lot when having access to the full dataset (validation MAE after cross validation around 9 compared to 6.94 when trained on the whole dataset). This is indeed a common feature of deep learning models: performances increase with the size of the dataset.