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

Group Leader: <**Mikheev**>

Acer Blake, Alexander Mikheev
{ab222@, am7522}@imperial.ac.uk
word count: 940

1 Part A

1.1 A-1 Tissue Segmentation

The segmentation model takes the form of a 3D U-Net which follows an encoder-decoder scheme. The encoder component consists of four levels of blocks, each with two convolutional layers, batch normalization, leaky ReLU activation functions, and one max pooling layer. The decoder follows the same outline, with the exception of using transpose convolutional layers. From the encoding layers, skip connections are used to bridge between the corresponding layers in the decoder.

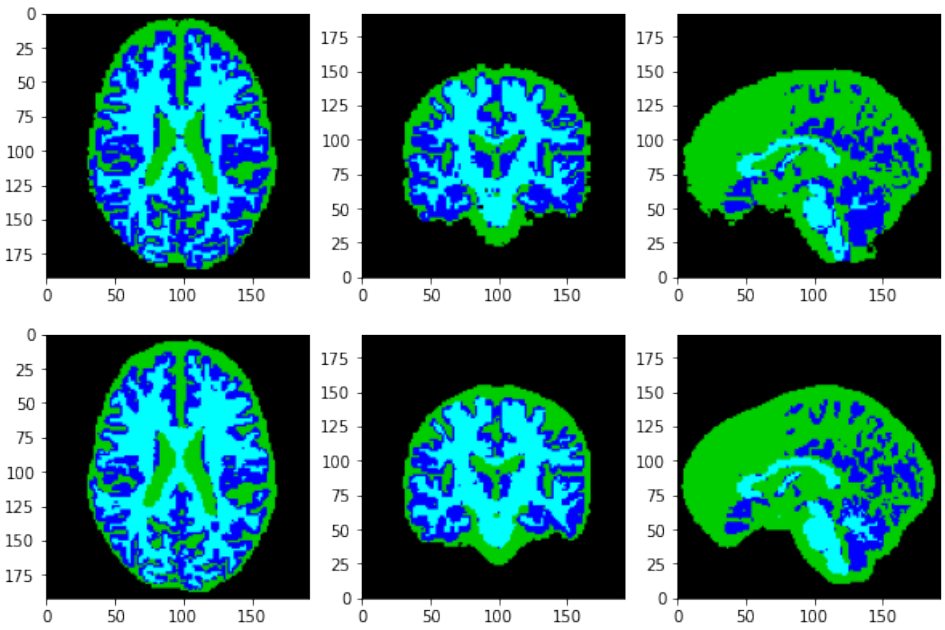

Figure 1: Ground truth segmentations (top) alongside model predictions (bottom).

All images were pre-processed using intensity normalization and resolution downsampling. A key change was to increase the resolution of the ground truth segmentation images from $64 \cdot 64 \cdot 64$ to $128 \cdot 128 \cdot 128$, which significantly improves the quality of the segmented images. An example of a subset of ground truth segmentations plotted against the model's predictions is given in Figure 1.

Hyperparameter tuning and the choice of optimisation algorithm played a critical role in maximising the performance of the 3D U-Net. These are summarised below in Table 1.

3D U-Net Hyperparameters	
Learning Rate	0.001
Batch Size	2
Epochs	30
Optimiser	AdamW

Table 1: 3D U-Net Hyperparameters

Using the aforementioned architecture we were able to generate very high quality segmented images, achieving a dice score of 0.92 and a converged loss of 0.77. These results are visualised in Figures 2 and 3 respectively.

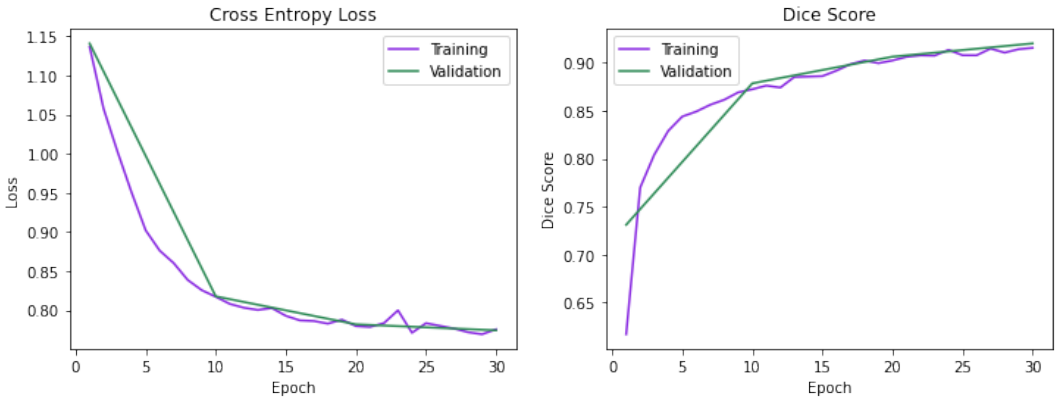

Figure 2: Cross-entropy loss and dice scores of the 3D U-Net model plotted against training epochs.

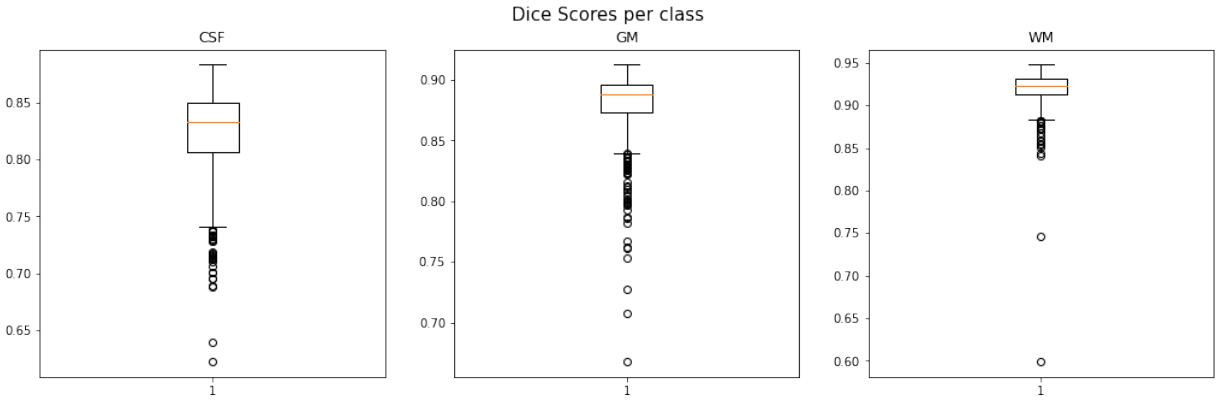

Figure 3: Box and whisker plot for class-wise dice score: cerebrospinal fluid (CSF), grey matter (GM) and white matter (WM).

1.2 A-2 Feature Calculation

The segmentation model from A-1 is performed on a test set of 500 images, and thereafter features for each tissue class are computed. The engineered features include the absolute tissue volume and relative tissue volume per class. However, on account of being significantly more discriminative per Figure 4, only the relative tissue volumes are used as input to the subsequent regression models.

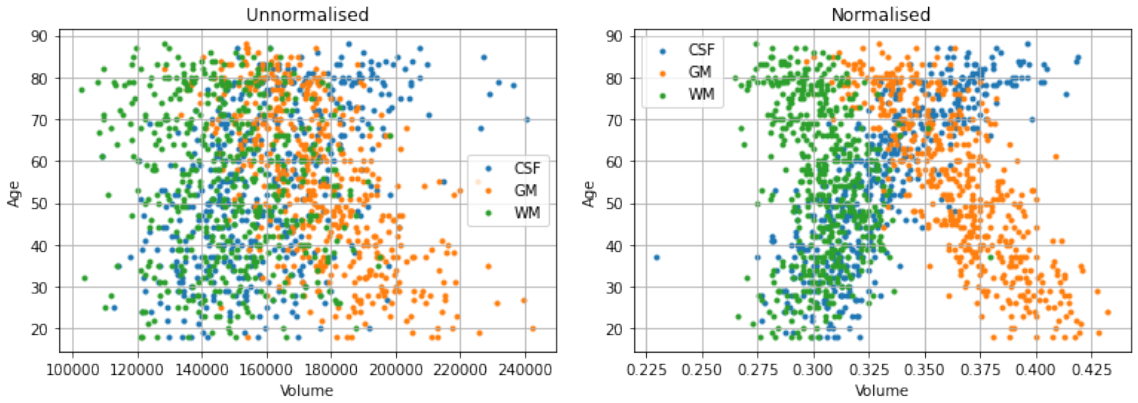

Figure 4: Unnormalised and normalised plots for cerebrospinal fluid (CSF), grey matter (GM) and white matter (WM) against total brain volume. Points are normalised in the right plot by taking the ratios between each tissue volume and overall brain volume.

1.3 A-3 Regression

The features are then fed into six regression methods: linear regression, ridge regression, lasso regression, k-NN, random forest regression, and support vector machines.

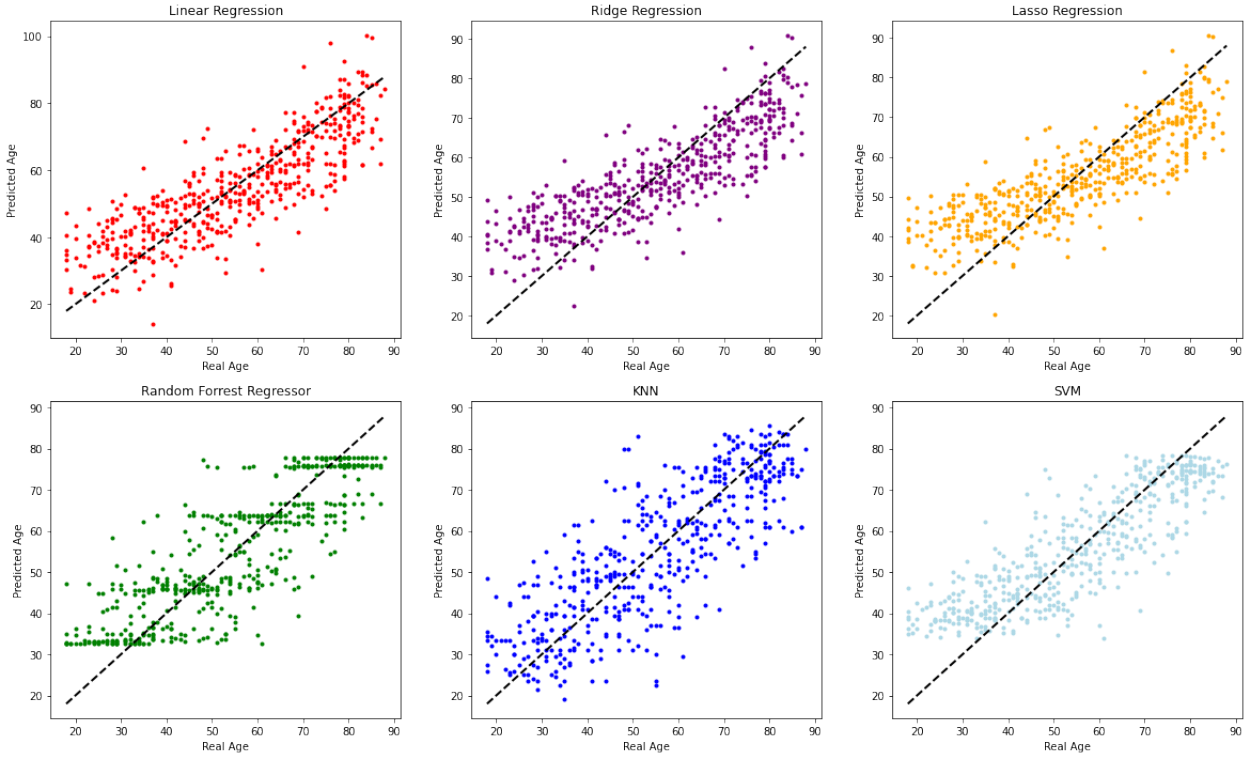

Figure 5: Individual regression plots for linear regression, ridge regression, lasso regression, random forest regression, k-nearest neighbours (k-NN), support vector machines (SVM).

Two fold-cross validation with random shuffling was used to train each regression model, with mean squared error, mean absolute error and R2 error metrics as the evaluation and performance criteria. The results, averaged across two-folds, are summarised in Table 2. These clearly indicate that random forest regression yields the best performance, and is thus selected as the final regression algorithm.

	MAE	MSE	R2
Linear	8.04	102.35	0.70
Ridge	8.64	114.55	0.66
Lasso	8.78	117.87	0.65
Random forest	7.78	97.11	0.71
KNN	9.04	132.01	0.61
SVM	8.02	99.1	0.71

Table 2: Predictive accuracy of various regression methods for task A-3.

1.4 A-4 Final Test on Hold-Out Data

The pipeline outlined in sections 1.1, 1.2, and 1.3 was recomputed on the 500 element hold-out training set, and then tested on the 100 element hold-out test set with the same hyperparmeters. This resulted in a final mean square error score of 70.84, a mean absolute error of 6.59, and an R-2 score of 0.81.

2 Part B

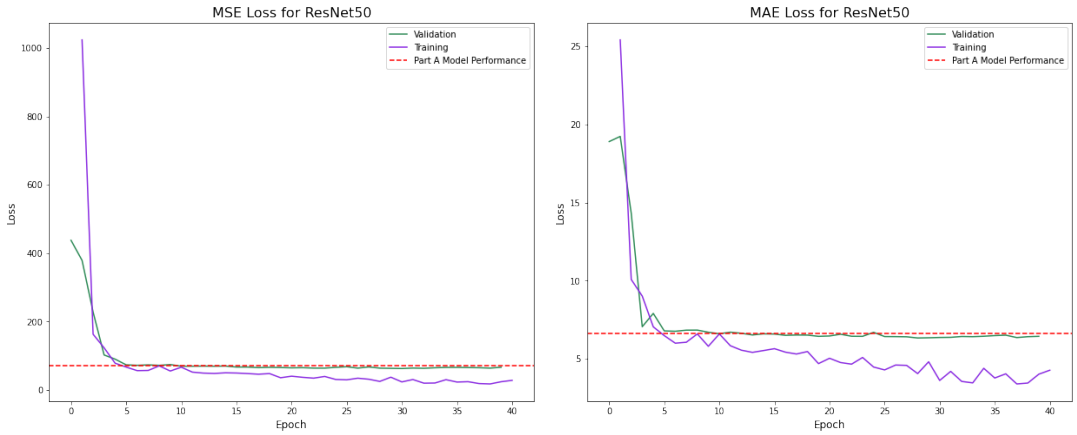

Figure 6: Training and validation loss curves for MSE (left) and MAE (right) as a function of the number of training epochs for the ResNet50 model, averaged over two-fold cross validation.

All images were pre-processed using intensity normalization and resolution downsampling to $64 \cdot 64 \cdot 64$. The initial approach to the task was a standard regression CNN consisting of convolutional layers, pooling layers, and ReLU activations, with a final series of linear layers, terminating with a single neuron outputting an unbounded scalar value. Through this approach our best-case MAE was ≈ 8 years. Increasing the model complexity with additional layers demonstrated improved performance, but significantly lengthened the training times and led to training instability. This motivated the transition to a more sophisticated ResNet architecture.

In addition to improving performance on the task through an increased fitting capacity, the ResNet architecture mitigates the problems associated with training deep convolutional neural networks by adding in residual connections, making it an appropriate choice for the present task. Regarding the details of the architecture, the model takes the form of a ResNet50, consisting of 50 convolutional layers distributed over 16 blocks, with ReLU activations. A final linear layer is appended at the end of the model to map the learned representation to a scalar output.

Batch normalization was incorporated to improve optimisation of the loss function, through standardizing the inter-layer outputs which stabilises the learning procedure [1]. The AdamW optimiser was selected over Adam through empirical experimentation, and has been shown to prevent overfitting by decoupling weight decay from the gradient update [2]. ReLu activation is a popular machine learning standard, effective through being computationally inexpensive and proved useful in mitigating the vanishing gradient problem [3]. Finally, incorporating weight decay was critical to achieving robust generalisation as the high model capacity of ResNet can easily lead to rapid overfitting.

ResNet50 Hyperparameters	
Learning Rate	0.001
Batch Size	16
Epochs	40
Optimiser	AdamW
Weight Decay	1e-5
Scheduler Decay Factor	0.1

Table 3: ResNet50 Hyperparameters

ResNet50 was retrained on the 500 subjects with two-fold cross validation. The training and validation loss curves are shown in Figure 6, demonstrating that both the training and validation MAE and MSE losses outperform the task A baseline. The final part B model produced a mean square error score of 60.34, a mean absolute error of 6.21, and an R-2 score of 0.84.

3 Age Regression Results

The final results reveal the ResNet architecture implemented for part B to outperform the random forest ensemble regressor for part A. These results are summarised in Table 4 below. Here we see that across all three computed metrics: MSE, MAE and R2 score, the ResNet outperforms the random forest.

Model	MSE	MAE	R2
A	70.84	6.59	0.81
B	60.34	6.21	0.84

Table 4: Final regression results of Part A and Part B on hold-out test set of 100 brain images.

The regression plots for tasks A and B are graphed below in Figure 7. While the plots do not show an unequivocally dominant regressor, we can observe that the random forest predictions show axis-aligned lateral dispersion along certain lines, whereas the nature of the dispersion along the line of best fit for the ResNetplot is more regular and dense. These results are consistent with our intuitions and the literature on the power of deep neural networks as function approximators. While ensemble methods such as random forests have been demonstrated to be highly robust, since each weak learner only possess a subset of the available features, learning complex relationships in data is harder to achieve. This is where deep neural networks thrive, and is reflected in the greater performance of the ResNet.

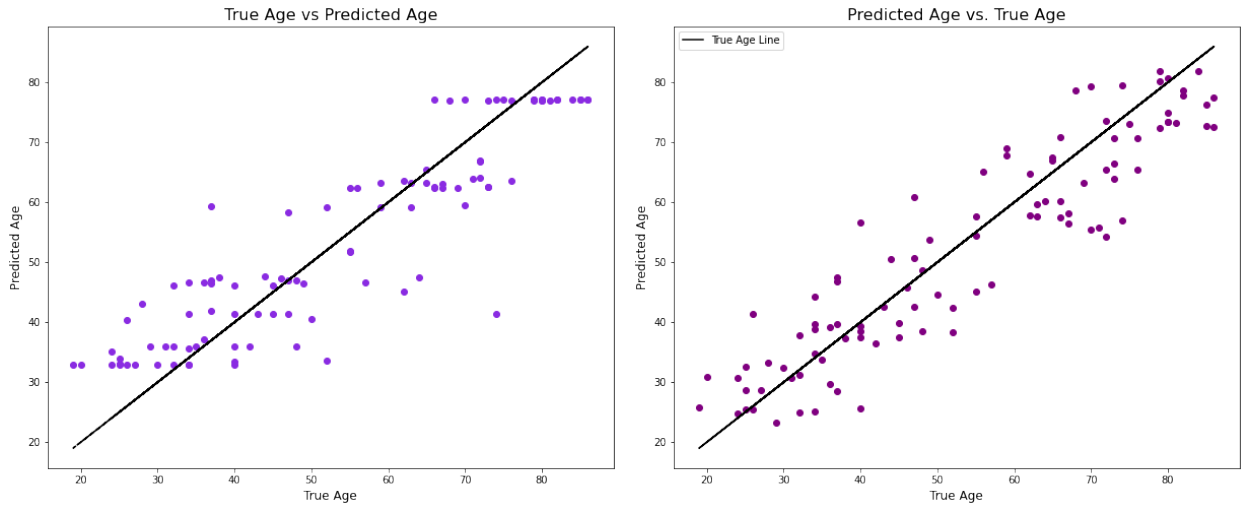

Figure 7: Scatter plots comparing the regression outputs of the random forest model (left) implemented for part A, and the ResNet50 (right) architecture implemented for part B.

A promising direction for further work would be to experiment with other state-of-art model architectures for segmentation, such as capsule networks.

References

[1] S. Ioffe, C. Szegedy, (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. ArXiv, abs/1502.03167.

[2] I. Loshchilov, F. Hutter, (2017). Decoupled Weight Decay Regularization. arXiv:1711.05101

[3] Agarap, A.F. (2018). Deep Learning using Rectified Linear Units (ReLU). ArXiv, abs/1803.08375.