DL Assignment 2: Brain Tumour Localization

Course - TI3155TU

Authors: Yan Tavares - 5559790

1. Data Preparation

The convolutional neural network (CNN) in highlight of this report aims to perform a localization box task of the cancer present in the MRI images. The dataset consists of 2146 Magnetic Resonance images, they are labelled in a ordered manner in a JSON file. Most images have resolution of 640 x 640. Not all the images are oriented as profile, some are from above and some seem to be from the top and some from the back.

To process the data into proper inputs and outputs for the convolutional network, it is necessary to create X tensors with shape (batch, channels, height, width) and Y tensors with shape (batch,outputs). Before generating those tensors, first a dataframe is created to map the outputs respective to the picture. Then the dataframe is shuffled with seed 42 using Panda's function "sample(frac = 1, random_state= 42)".

Three sections of the dataframe are made such that the first 50% entries are designed for training, the next 20% for validation and the last 30% for testing. The jpeg image files are transformed to image objects by means of PIL package. By accessing the dimensions of the image object it is possible to normalize outputs respective to it. Table 1 show an example of the non-normalized and the normalized y target values for image 1294.jpg. A batch tensor of normalized outputs is named as Y_{bbox}.

Table 1. Normalized and non-normalized target values of image 1294.jpg. Non-normalized values are divided by 640 which is the maximum resolution of the picture in specific.

	x_{box}	y_{box}	w	h
Non-Normalized	253.0	233.0	97.5	83.75
Normalized	0.3484	0.4469	0.1289	0.1309

The image files are reshaped to 300x300 and transformed to batch tensors X. Each pair of batch tensors X and Y_{bbox} are stored in a tensor file "tensors_batch_{i}.pt". Those tensors are stored in three different folders; train, val and unseen. By storing in separated folders it improves organization and ensure only training data is read for training. If desired, while training, some specified fraction of the X[i] tensors can be augmented, such that the image and the bbox are transposed and/or noise is added. Augmentation can reduce overfitting and perhaps allow the model to learn more. Figure 1 shows an example of a augmented image.

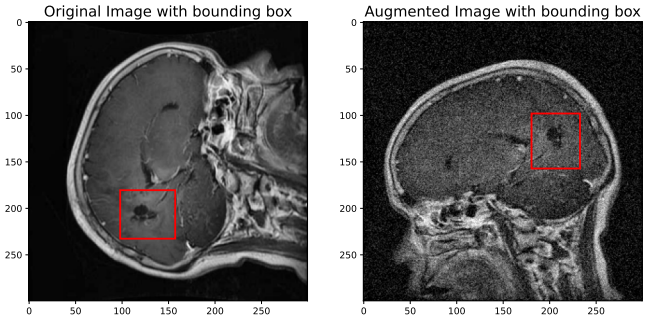


Figure 1. Comparison between augmented and non augmented images and its respective bounding box

2. Model Design

2.1 Loss function

Three types of loss function were tested while training the model, mean squared error (MSE), mean absolute error (MAE) and SmoothL1Loss from troch. The test loss is stored per batch while the validation loss is stored per epoch. During training, the validation is only stored for reference and not used as stopping criteria. The performance measure is used as stopping criteria instead.

2.2 Performance metric

The performance metric used is intersect over union (IoU), the computation is done per tensor batch to increase speed. This metric computed as showed in Equation 1, and is used for the early stopping criteria.

$$\text{IoU} = \frac{\text{intersec\_area}(Y_{bbox}, Y_{hat})}{\text{area}(Y_{bbox}) + \text{area}(Y_{hat}) - \text{intersec\_area}(Y_{bbox}, Y_{hat})} \quad (1)$$

2.3 Model choice

The CNN was trained from zero and has a architecture very similar to AlexNet, the number of layers, type of layers, kernels, strides and paddings are the same. The number of hidden channels for each layer is customized such that memory usage is reduced. Check the originally made CNN diagram in Figure 2 for a complete overview of the model architecture.

The initial weights are set using Xavier uniform with torch in determinist mode and seed 84. Three different types of optimizers can be used, Aadam, AdamW and Adragrad. Regularization L1 and L2 can be used to reduce overfitting and possibly allowing the CNN to keep training for longer.

2.4 Hyperparameter selection

There are innumerous hyperparameters in this model, but the tuning was only performed on a few of them due to infeasibility of attempting

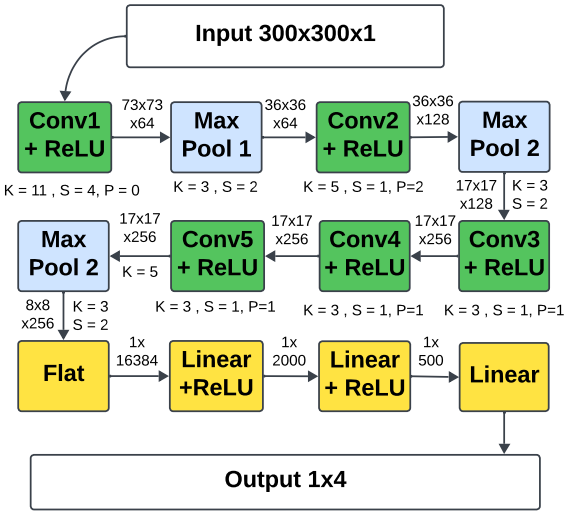


Figure 2. Caption

every possible combination. By performing initial guesses and runs, a range of potential values for each hyperparameters was made. Table 2 shows the parameter grid carefully evaluated for this CNN.

Table 2. Grid of potential hyperparameters tested during tuning

Parameter	Options
Learning Rate	$[10^{-4}, 10^{5}, 10^{-6}]$
Batches per epoch	[100, 50, 30, 20]
Loss criteria	[MSE,MAE,L1smoothloss]
L1 Reggularization	$[10^{-5}, 10^{-6}, 10^{-7}, 0]$
L2 Reggularization	$[10^{-5}, 10^{-6}, 10^{-7}, 0]$
Augmentation Ratio	[0.5, 0.3, 0.1, 0]
Optimizer	[Adam, AdamW, AutoGrad]

Based on the IoU results in the validation set, the best hyperparameter selection is:

```
{Learning Rate : 10**(-5),
Batches per epoch: 30,
L1 Reggularization : 0,
L2 Reggularization : 0,
Augmentation Ratio: 0,
Optimizer: AdamW}
```

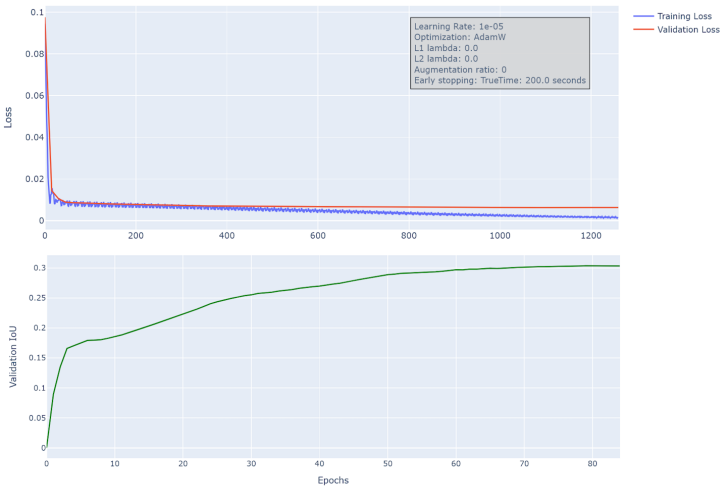
2.5 Baseline choice

A dummy model was trained in the training set and then tested in the unseen set to be used as the baseline performance. This model always predicts the average values of x_{box}, y_{box}, w and h present in the training set.

3. Results

3.1 Training Curves

Most of the curve characterizes a proper fitting training. From epoch 70 it is possible that the validation loss starts increasing while the training loss decreases characterizing a standard overfitting, however IoU still increases from 0.3000 until to 0.304 at epoch 80. The possibility of capturing this small improvement can only happen if IoU is used as the stopping criteria.



3.2 Inference

To visualize the quality of the predictions, 6 pictures are inspected in Figure 3. The predictions are reasonably good, in all the six cases the predicted box is at least partially inside the ground truth box. There is a issue however, as can be seen in image 1130.jpg and many others present in the dataset, a considerable portion of ground truth boxes completely misses the tumour. This misplaced ground truth boxes can strongly confuse the CNN and strongly limitate the performance.

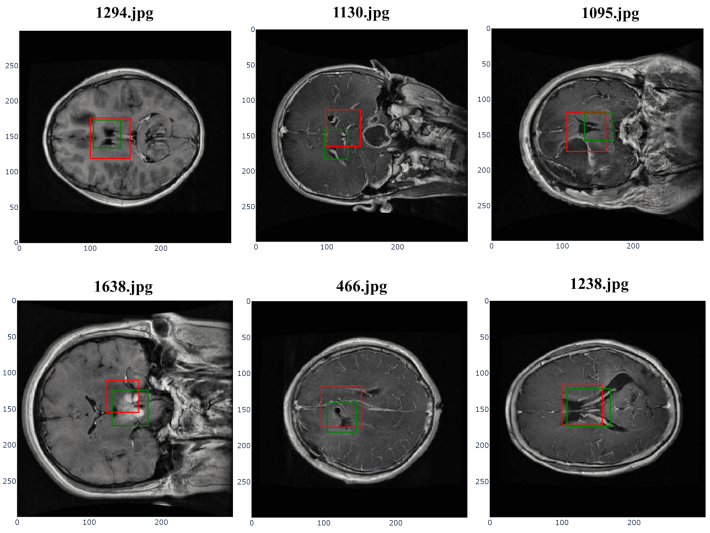


Figure 3. Inference of predictions for six different images. The ground truth box is green and the predicted box is red.

4. Conclusion

Table 3. Performance metric comparison of tuned model and baseline

Model	IoU on unseen set
Tuned CNN	0.3051
Dummy	0.182

The CNN evaluated in this report has a reasonable performance considering the faulty data present on the dataset. The values of IoU for the validation set and unseen set which means it generalized well to unseen data even without making use of data augmentation and regularization.

During training it was noted that MSE loss criteria could mitigate the issue of the CNN ignoring certain types of tumour and making predictions completely far from the ground truth. It had a very significant impact, elevating IoU from 25% to 30%, but worked the best in combination with 30 batches per epoch. SmoothL1Loss criteria performed very similarly to MSE loss. MAE loss had the worst performance.

There wasn't a significant difference in performance switching optmizer between Adam and AdamW, but AdamW performed better in general. AutoGrad optimizer took longer to converge and performed worse than Adam and AdamW.

Regularization and data augmentation did not improve results, it makes the IoU decrease by 1 or 2%. This might have happened because the faulty data is already confusing the network and generalizing predictions. The loss gradient field for each batch with faulty data changes from the ideal making the model step away from what it needs to learn, consequently generalizing the predictions. The addition of regularization and data augmentation only over-generalizes the network.

From the results of the grid-search, it is not expected that this architecture trained from scratch would produce much better results than 30%. To improve performance it is recommended to filter the faulty dataset, increase the CNN deepness and number of hidden channels.