02514 Deep Learning in Computer Vision

Project 2

Kim Reinhardt Jensen, s164038
Michael Rahbek, s164035
Ghassen Lassoued, s196609

Technical University of Denmark
June 26, 2020

1 Dataset

In this project we work with the LIDC-IDRI segmentation dataset. It consist 2D patches from CT lung scans where each patch has lesions annotated by four different radiologists. The dataset is organised into a training set of 8843 images, a validation set of 1993 images and a test set of 1980 images. The images are already downsampled to 128x128 with pixel values between 0 and 1 and we use these specs as input. In Section 1-4, only the first annotated lesion image is loaded for evaluation. In Section 5, all four annotated lesion images are used in an ensemble model.

2 SegNet

In Exercise 2 a simple segmentation CNN with a SegNet style architecture was implemented and this served as our starting point. The segmentation network starts with an encoder-part which downsamples the input image from spacial dimensions of 128 to spacial dimensions of 8. It starts with a 3x3 convolution with padding 1 that increases the number of channels from 1 to 64. Then a 2x2 maxpool with stride 2 halves the spacial dimensions. From here, 3x3 convolutions keeping 64 channels with padding 1 followed by 2x2 maxpools with stride 2 is performed until the spacial dimensions is 8. Here a 3x3 convolution with padding 1 is the so-called bottleneck layer of the model. The bottleneck layer is a compressed lower dimensionality representation of the input containing the most important features it detects from the input. This is then used in the decoder-part of the network which upsamples the spacial dimensions back to 128 to output a lesion segmentation. In the decoder, upsample is used to double the spacial dimensions and then a 3x3 convolution with padding 1 is performed keeping the 64 channels. This is repeated until the spacial dimensions are 128 and the last 3x3 convolution reduces the output channel to 1. ReLU is used as the activation function after all the convolutional layers.

Convolution and transpose-convolution can be used instead of maxpool and upsample. However, this increases the number of parameters by about 30 percent, and since it had no significant performance improvement, it was omitted. In the training we used the Adam optimizer which was tested with different learning rates, and a learning rate of 10^{-4} showed to be the best performing one. The batch size was set to 64 to better utilize the power of the GPU. There was experimented with different loss functions and the focal loss was the best performing which can be explained by its ability to handle class imbalances. The idea is, that the pixels with higher uncertainty from the output classification is weighted more than the pixels where the model is very certain of the output class. This is important as there is a class imbalance of $[167 : 1]$ in the training set and most pixels from the negative class are thus 'easy' to classify.

2.1 Performance evaluation

Because of the significant class imbalance, the choice of evaluation metric is important when measuring the performance of our segmentation networks. The pixelwise accuracy is obviously not a suitable metric as it is possible to get a high accuracy by predicting only the negative class, i.e. outputing a black image. A better metric would be the Intersection over Union or Jaccard index. The advantage of this metric is that the network cannot achieve a good performance by only considering the majority class. The dice overlap is similar to the IoU though IoU tends to penalise wrongly classified pixels greater. The weakness of these two metrics becomes apparent when considering a classification task with only one pixel of the 'positive' class. In this case the classifier will receive a relatively low score for both IoU and dice if it just includes a few false positives.

The sensitivity is a measure of how well the model predicts the segmented positive class and the specificity measures how well the model predicts the non-segmented negative class. Specificity suffers from the same problem as the accuracy as this can easily get a good measure predicting

everything as non-segmented in an imbalanced dataset. The sensitivity has another pitfall as it is large if the output predicts a huge segmented area encapsulating a small real segmented area.

As is clear from the above overview, the choice of evaluation metric depends both of the class imbalance in the data and the goal of the segmentation task. In cases where it is important to correctly capture every member of the minority class even if this results in quite a few false positives, the sensitivity is very well suited. If a more 'balanced' prediction is the goal, then the dice overlap or the IoU would be reasonable choices.

In Table 1 the focal loss and the five performance metrics for the SegNet style architecture is shown for the training, validation and test set. It is seen that the specificity and accuracy are similar and close to 1 as expected from the large class imbalance, hence they are unsuitable for the task. As the model does not classify too many pixels in the positive class, sensitivity is a useful measure to look at here, but should still be carefully interpreted. The best overall measures are probably the dice and IoU as they show how well the model predict the segmented positive class. As expected it is seen that the IoU penalizes wrongly classified greater than dice, and the model performs best for the training set and worst for the test set.

Split	Loss	Accuracy	Dice	IoU	Sensitivity	Specificity
Train	0.001004	0.9933	0.5543	0.3848	0.8476	0.9993
Validation	0.004036	0.9922	0.3397	0.2167	0.5269	0.9983
Test	0.005887	0.9919	0.3209	0.2016	0.5104	0.9980

Table 1: Evaluation metrics for the three splits using the SegNet style architecture.

3 U-Net

We first tested the U-net architecture used in exercise 2, where all layers had 64 channels. The networks consisted of four 2x2 maxpoolings with a 3x3 convolution between each of these. In the bottleneck, we had a single 3x3 convolution and the spacial dimensions of the image was 8. In the decoder-part of the network we had four up-samplings with a 3x3 convolutions between each which resulted in an output image with 1 channel and spacial dimensions of 128 pixels. Therefore, the network is very similar to our SegNet architecture with the skip connections included. Using the focal loss, we were able to get some useful predictions. At first, we tried using the regular binary cross entropy as loss function but this caused all of the output predictions to be black which this is seen to give a relatively low loss because of the class imbalance. We then used the focal loss, which gave good results as in the case of SegNet.

3.1 Performance evaluation

In Table 2, the focal loss and the five performance metrics for the U-Net are shown for the train, validation and test set. As discussed above, dice overlap and IoU are good overall measures of performance, while the accuracy and specificity do not contain much useful information. And again, the model performs best on the training set and worst on the the test set.

Split	Loss	Accuracy	Dice	IoU	Sensitivity	Specificity
Train	0.000858	0.9934	0.5873	0.4172	0.9394	0.9990
Validation	0.004765	0.9918	0.3721	0.2416	0.6381	0.9976
Test	0.006936	0.9914	0.3380	0.2150	0.6276	0.9972

Table 2: Evaluation metrics for the three splits using the U-Net style architecture.

In Figure 1 the five performance metrics are shown during the training of the network for the training and validation set. It is seen that the accuracy changes relatively little throughout. The IoU and dice overlap are seen to be strongly correlated. The sensitivity grows throughout the training while the specificity stays roughly constant just below 1.0. When compared with the loss in Figure 2, overfitting is not observed for these metrics as they are not used for optimisation.

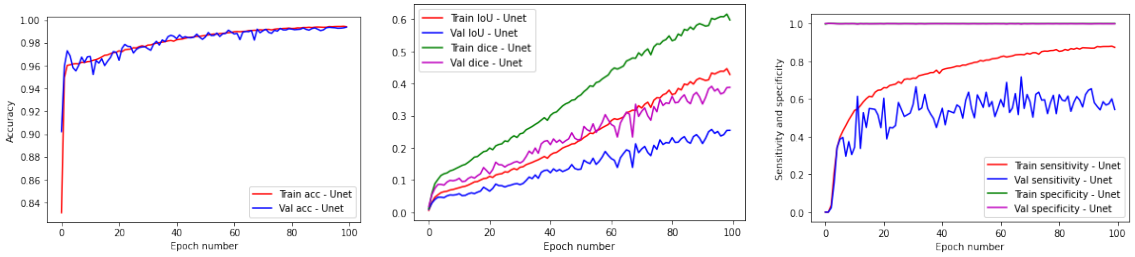

Figure 1: The five performance metrics for the U-net evaluated on the training and validation set. Please note 'dice and IoU' and 'sensitivity and specificity' are shown in the same plots.

4 Improving the segmentation network

The simple transformation of the SegNet to the U-Net architecture improved the performance of the segmentation. This is seen from the results from Table 1 and 2 where the U-Net performs better measured on the validation set for both the dice and IoU metrics. Therefore to further improve the segmentation we focused on developing the U-Net.

First there was experimented with increasing the positive weights of the binary cross entropy, which has the effect penalising wrong predictions of the positive class more. We tried to combine this with the focal loss and naturally, this resulted in more white pixels, i.e. more frequent predictions of the positive class. Depending on the metric this could be considered an improvement but since this change generally generates significantly more false positives, we consider this change to worsen the model, as we already use the focal loss to handle the class imbalance.

We also tried with regularization, specifically we added sparsity as described in Exercise 2. This was done because the output from the network seem to predict too large areas with lesions, where much of the false positives were small patches around the true lesion. Some of these extra patches also looks like remnants from the convolutions. The regularisation was implemented by augmenting the focal loss with the sparcity: $\mathcal{L}_{loss} = \alpha_{reg}\mathcal{L}_{focal} + (1 - \alpha_{reg})\sigma(\hat{y})$

After experimenting we chose $\alpha_{reg} = 0.9$. The learning curves for the original model and the two modifications is seen in Figure 2. Measured on the focal loss for the validation set, the sparcity regularization is the best performing, which is due to the successful removal of the small patches. The results for the models segmentation for one image instance is shown below in Figure 3. Here it is seen that the positive weight increases the number positive class pixels of the output and the sparcity regularisation reduces the amount of white in the output image.

Another way we tried to improve our U-Net was to modify the network. We changed the architecture to be more similar to the one given by the paper Olaf Ronneberger and Brox 2015. The important change was to increment of the number of channels towards the bottleneck layer instead of keeping the number of channels at 64. The idea is to gain more useful information to the bottleneck layer which is then used by the decoder to reconstruct the segmentation. We furthermore implemented batch normalization, changed the upsamplings to transpose-convolutions and used a 1x1 convolution at the end of the network as a fully connected layer. This architecture reduced the total number of parameters by about 30 % compared with the previous Unet archi-

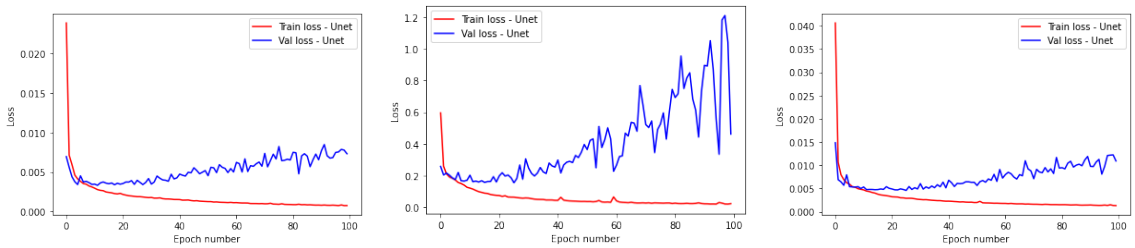

Figure 2: Learning curves for the the U-net. The left figure: Initial U-Net. the middle figure: With positive weights. The last figure: With sparcity regularisation with $\alpha_{reg} = 0.9$.

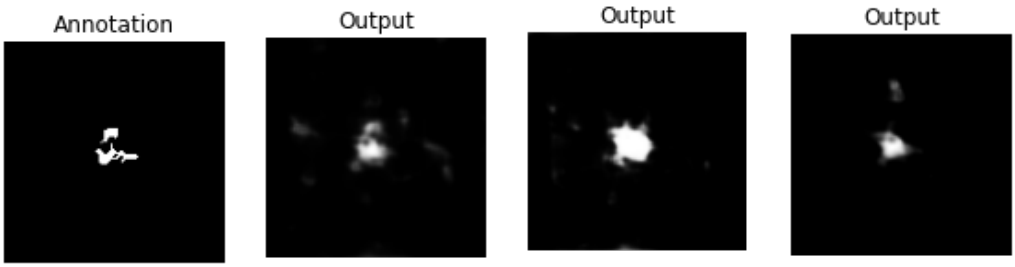

Figure 3: Sample image from the validation set trained on the U-net. The second image from the right uses positive weights and the rightmost image uses sparcity regularisation with $\alpha_{reg} = 0.9$.

tecture. The comparison of the two networks would be more fair if they had the same number of parameters but we did not have time to test this.

In Table 3 the five performance metrics for the U-Net with positive weights, U-Net with sparcity regularization and U-Net with the updated architecture is shown for the train, validation and test set. In this table we see the problem of the sensitivity as this favours the U-Net with positive weights. However, the dice and IoU shows this is actually the poorest performing of the three. Measured on dice and IoU, the U-Net with sparcity regularization is the best performing network, and the network we use in the next section for uncertainty evaluation.

Apart from adjusting the loss function, applying regularization and changing the architecture, the segmentation may be improved by using data data augmentation to reduce overfitting. Data augmentation was not implemented in this project, but if we were to apply it in future works with this dataset, it is important that identical transformations are applied to both the 2D patch and the segmented lesion in the training set.

metric	U-net, pos. weights			U-net, regularisation			2nd U-net, regularisation		
	Train	Val	Test	Train	Val	Test	Train	Val	Test
Acc.	0.9895	0.9886	0.9887	0.9968	0.9959	0.9952	0.9964	0.9952	0.9948
Dice	0.5242	0.3708	0.3449	0.7158	0.4639	0.4201	0.6944	0.4066	0.3653
IoU	0.3563	0.2389	0.2178	0.5579	0.3146	0.2812	0.5327	0.2689	0.2374
Sensi.	0.9995	0.8504	0.7751	0.8453	0.5161	0.4685	0.8789	0.4817	0.4450
Speci.	0.9927	0.9924	0.9928	0.9996	0.9989	0.9985	0.9996	0.9988	0.9986

Table 3: Evaluation metrics for the three splits and the three versions of U-net.

5 Segmentation uncertainty

We constructed an ensemble network using the U-Net with 64 channels in all layers and regularisation as described in Section 4. We trained this four times using the four different annotations. By combining the predictions from these four networks, we are able to quantity the uncertainty related to the annotations from the experts. This uncertainty is related with the general difficulty in the detection of lesions and the preferences of the annotators. In fact, the 4 radiologists who made this annotation are chosen out of 12 , and it is not clear whether the same 4 radiologists always provided the annotations. The uncertainty comes also for the experts preferences: even if the annotations were made independently, these annotations vary from an expert to another. The prediction from this network on a single image is shown in Figure 4 with mean probability and standard deviation. Below in Figure 5 is shown the same for the annotated lesions.

In addition to this uncertainty, there is also a numerical uncertainty connected with the network and general noise. The probabilistic U-net tries to quantify the variability for the individual image by augmenting the network with a variational auto encoder that models the probability distribution in the latent space and draw samples from this in the prediction. Another way of describing this uncertainty is by using dropout in both the training and evaluation phase. In this way, the same network may be used to output several predictions for a single image, which may be used to estimate a probability distribution.

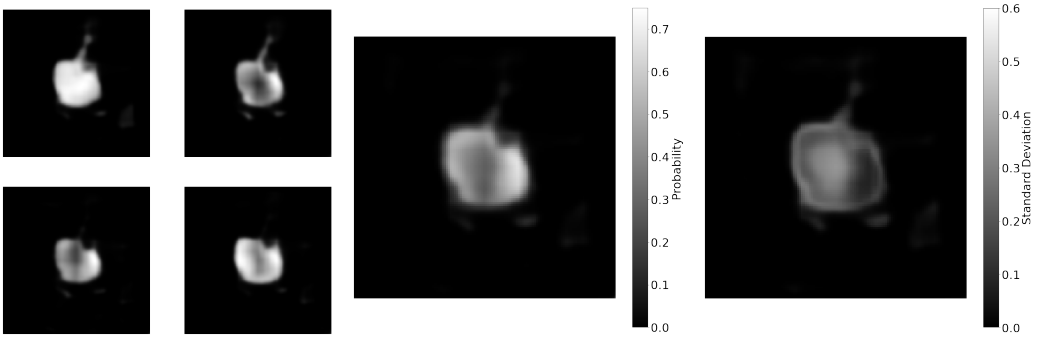

Figure 4: The four images on the left shows the individual predictions from the models, while the figure in the middle shows the mean and figure to the right shows the standard deviation.

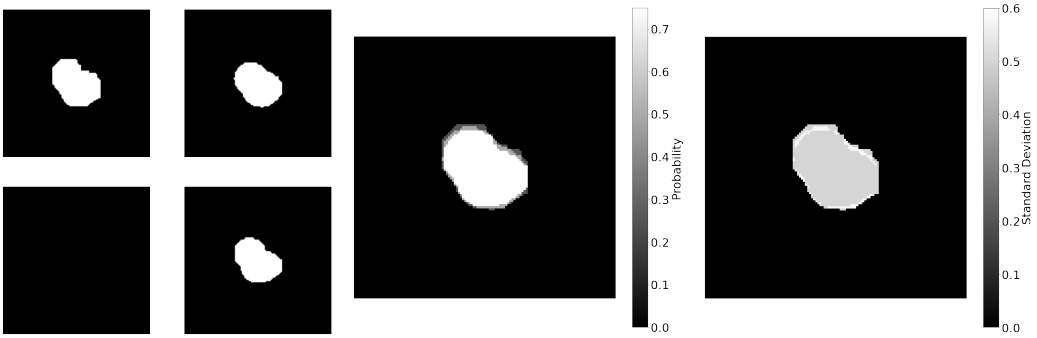

Figure 5: The four images on the left shows the individual annotated lessions, while the figure in the middle shows the mean and figure to the right shows the standard deviation.

References

Olaf Ronneberger, Philipp Fischer and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In:

Diary

Monday and Tuesday

- Kim and Michael have been present from morning to evening and worked together on all parts on the report and the coding of the models. We had implemented the models and written a bit on the report before group 19 and 20 was merged.
- Grimur and Ghassen: data preparation, implementing loss functions, implementing SegNet (in group 19)

Wednesday and Thursday

Ghassen joined the group 20 on Wednesday. He was briefed on the process and the code. As the code was pretty done, Kim generated plots and results. All three of us worked on the report.