TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Biomedical Engineering

Project imaging

GitHub repository name of the group: https://github.com/poppywalk/Project-Imaging-Group-3

Assignment 4

Laslo van Anrooij	1264109
Kirsten Lukassen	1228126
Maud van Ratingen	1257838
Iris van der Werf	1326058

February 25, 2021

---

1 Exercise 1

1.1 When does transfer learning make sense? Hint: watch the video. Does it make sense to do transfer learning from ImageNet to the Patch-CAMELYON dataset?

Transfer learning from ImageNet and Patch-CAMELYON only makes sense when both of the tasks it needs to do have the same input. ImageNet needs to have a lot more images than the Patch-CAMELYON dataset to work well. Transfer learning makes sense when you suspect that low-level features from the pre-trained model could be helpful for learning the new model (Deeplearning.ai (2017)).

ImageNet and Patch-CAMELYON both use images as a input. This means transfer learning can be used, because the input for the tasks is the same. The dataset from ImageNet is very large with about 14 million images (ImageNet (2010)), but it does contain features which are not easy to spot. It does its image recognition based on colour and contour. In histopathology, more things weigh in on the chance of something containing metastases, so when using the ImageNet data some fine-tuning is needed in a model. The data set from Patch-CAMELYON can provide this fine tuning, because it is a smaller data set (Patch-CAMELYON (2021)).

2 Exercise 2

2.1 Run the example in transfer.py. Then, modify the code so that the MobileNetV2 model is not initialized from the ImageNet weights, but randomly (you can do that by setting the weights parameter to None). Analyze the results from both runs and compare them to the CNN example in assignment 3.

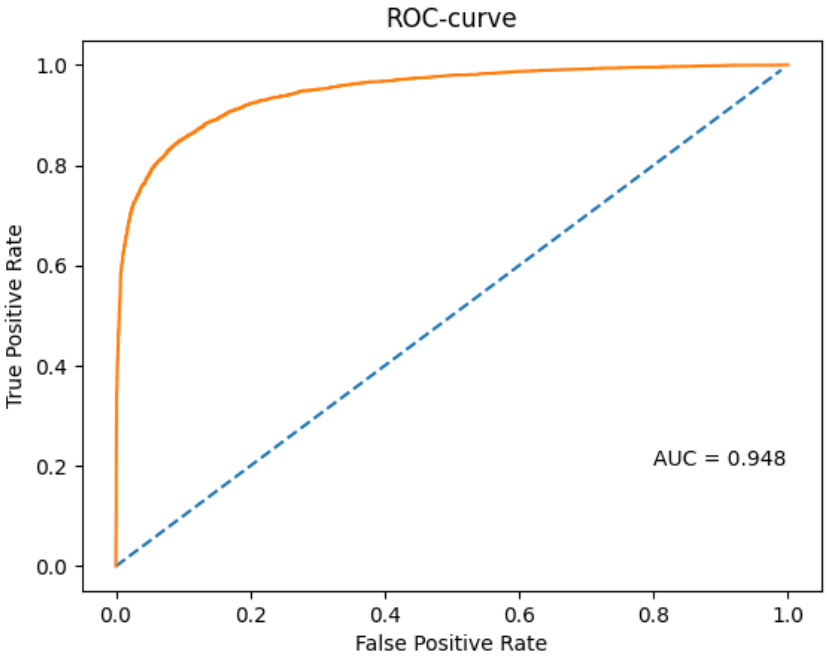

Figure 1: The ROC-curve with ImageNet weights

When using the ImageNet weights the AUC has a value of 0.948. This means that the model does a fairly good job considering an AUC of 0.0 gives that the predictions of the model are true for 0% and that an AUC of 1.0 gives that the predictions of the model are true for 100%.

The code can be found in Assignment-4-Group03.py.

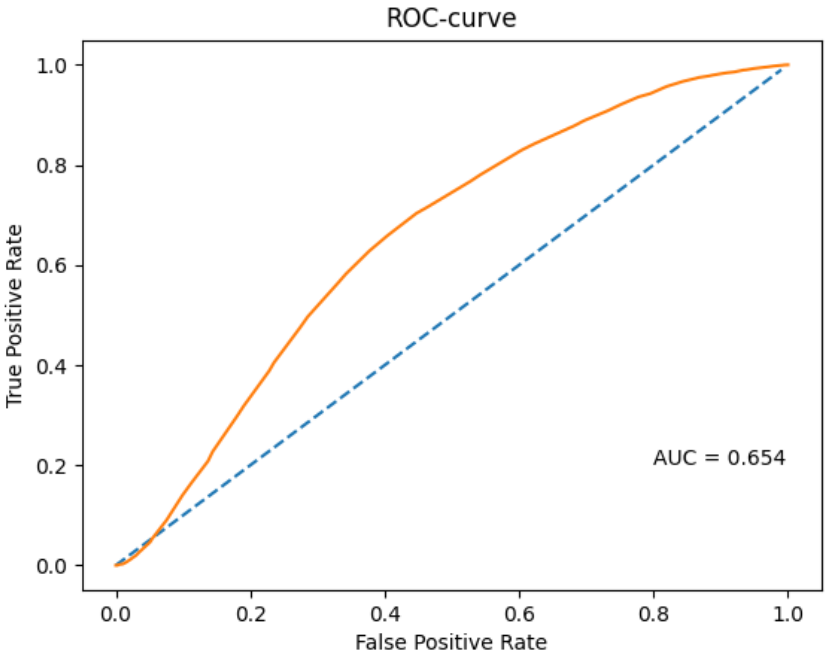

Figure 2: The ROC-curve without ImageNet weights

When not using the ImageNet weights the AUC has a value of 0.654. This model preforms a lot worse then the model with the ImageNet weights (which had an AUC of 0.948).

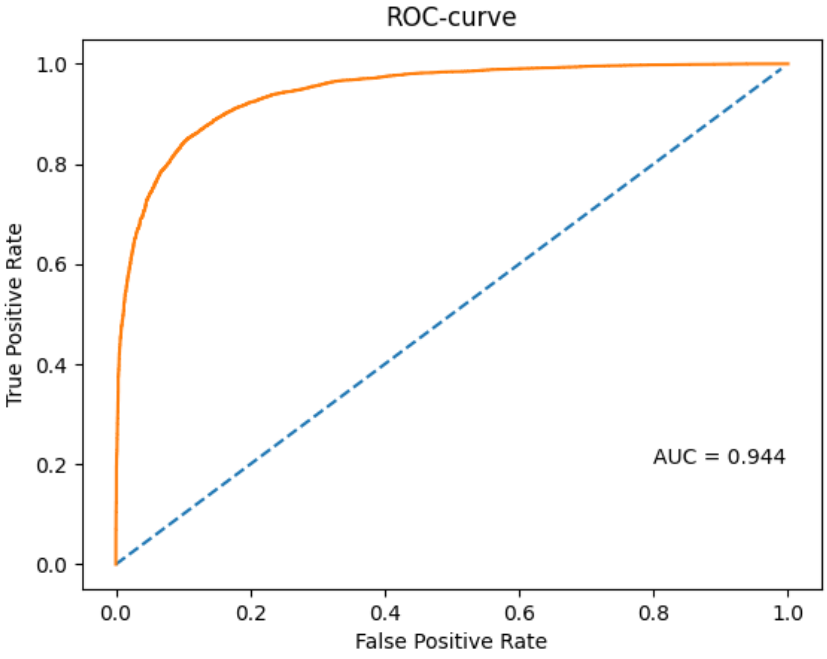

Figure 3: The ROC-curve CNN model

The CNN model of assignment 3 had an AUC of 0.944. This means that does a fairly good job. The model with the ImageNet weights has a higher AUC than the CNN model. The model without the ImageNet weights is trained on a lower amount of images then the model with the ImageNet weights effectively. This means that the model without ImageNet weights is more prone to over fitting and getting in a local minimum. This is probably what happened during training and can explain the low AUC. The model with the ImageNet weights preforms slightly better then the CNN model, this could be because the model has been trained on more images then the CNN model effectively.

3 Exercise 3

3.1 The model in transfer.py uses a dropout layer. How does dropout work and what is the effect of adding dropout layers the the network architecture? What is the observed effect when removing the dropout layer from this model? Hint: check out the Keras documentation for this layer.

Large neural networks trained on small data sets can over-fit on the training data. This results in a poor performance when the model is used on new data, like the test set (Srivastava et al. (2014)).

The problem of over-fitting can be dealt with by using a dropout layer. In a dropout layer randomly sets input units to zero with a frequency of the argument 'rate' at each step during training time. The input units which are not set to zero are scaled with 1/(1-rate) to keep the sum of the inputs the same as before (Srivastava et al. (2014)). The network can now with every step look at a combination of features instead of just one feature, which reduces the over-fitting (Goodfellow et al. (2016)). The dropout-layer prevents over-fitting, which means that the validation loss will most likely increase when the drop-out layer is removed.

3.1.1 Removing the dropout layer with the ImageNet weights

The table (table 1) below shows the validation loss before and after the dropout-layer is removed. It also shows the validation accuracy before and after the dropout-layer is removed. The model was trained with the imagenet weights. The code can be found in Assignment-4-Group03.py.

Validation loss before	Validation loss after	Validation accuracy before	Validation accuracy after
0.26520	0.26421	0.9087	0.8963

Table 1: The validation loss and validation accuracy before and after removing the dropout-layer with the ImageNet weights

As can be seen in the table above (table 1) the validation loss decreases with 0.00099 and the validation accuracy decreases with 0.0124 when removing the dropout layer. The validation accuracy and validation loss do not change a lot when removing the dropout layer. This could be due to the transfer learning. The transfer learning could have learned the base features for classification, leaving less room for over fitting in the final layer. Therefore the dropout layer does not have a large effect on both the validation accuracy and the validation loss.

When inspecting the validation loss curves (as can be seen in figure 4 and figure 5 below), no notable differences can be seen when comparing the difference between training (in orange) and validation (in blue) loss trajectories in each curve. Therefore there is no hard evidence that the dropout layer prevents overfitting in this case as opposed to what is expected based on the information in Goodfellow et al. (2016). The effect of the dropout layer may be more noticeable in larger networks or with smaller data sets or when not using

the ImageNet weights. This last possibility will be discussed in the next section.

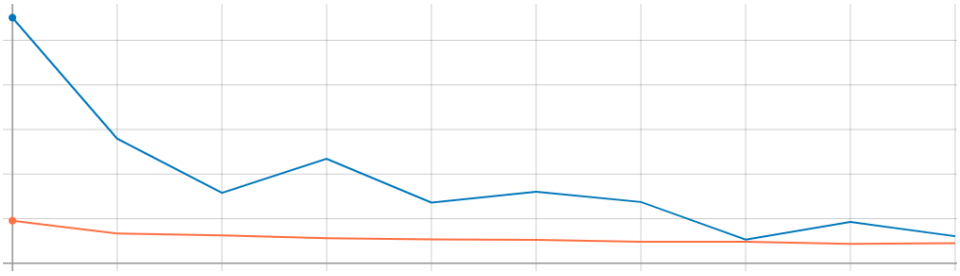

Figure 4: Validation loss (blue line) and training loss (orange line) with the dropout layer with weights set to imagenet

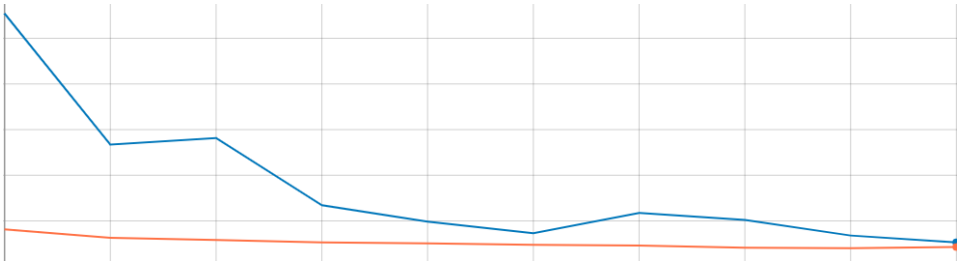

Figure 5: Validation loss (blue line) and training loss (orange line) without the dropout layer with weights set to imagenet

3.1.2 Removing the dropout layer without the ImageNet weights (weights set to None)

The table below (table 2) shows the validation loss before and after the dropout-layer is removed. It also shows the validation accuracy before and after the dropout-layer is removed. The model was trained without the ImageNet weights. The code can be found in Assignment-4-Group03.py.

Validation loss before	Validation loss after	Validation accuracy before	Validation accuracy after
0.69151	0.69278	0.5350	0.5138

Table 2: The validation loss and validation accuracy before and after removing the dropout-layer without the ImageNet weights

As can be seen in the table above (table 2) the validation loss increases with 0.00227 and the validation accuracy decreases with 0.0212 when removing the dropout layer. These values are not very large, but, even when the changes are small they are in line with the information in Goodfellow et al. (2016). When looking at the validation loss and training loss curves in the figures below (figure 6 and figure 7), the effect of the dropout layer can be seen.

In figure 6 (with the dropout layer) some overfitting occurs, the training loss and validation loss curves are far apart and do not follow the same trajectory. This could be due to the weights which have been set to None. When the weights are set to None, there is to little data for the model to learn the features in the images. Which results in overfitting. When looking at figure 7, the difference the dropout layer makes can be seen (here there is no dropout layer). The model starts overfitting a lot. The validation loss (in blue) even goes up and does not follow the training loss (in orange). This means the dropout layer does indeed improve the model by making it overfit less.

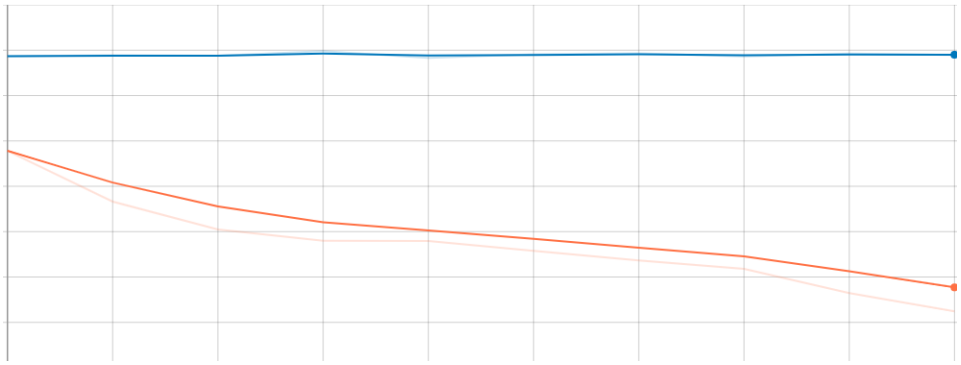

Figure 6: Validation loss (blue line) and training loss (orange line) with the dropout layer with weights set to None

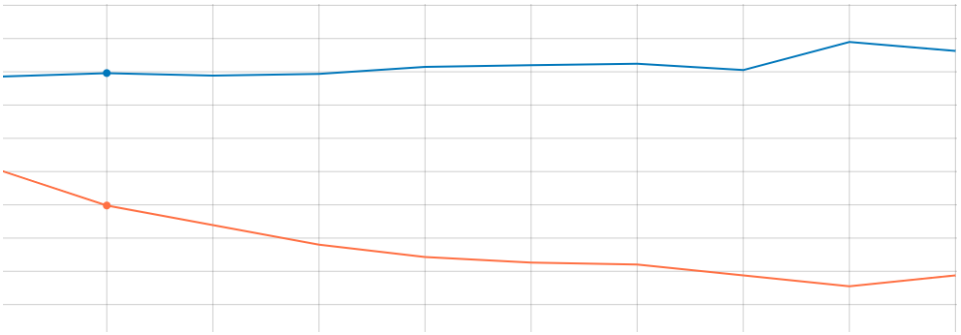

Figure 7: Validation loss (blue line) and training loss (orange line) without the dropout layer with weights set to None

References

Deeplearning.ai (2017). Transfer learning (c3w2l07) [youtube].

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

ImageNet (2010). About imagenet[summary and statistics (updated on april 30, 2010)]. http://www.image-net.org/about-stats,.

Patch-CAMELYON (2021). Catalog[patch-camelyon (updated on 2021-01-28)]. https://www.tensorflow.org/datasets/catalog/patch_camelyon.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.