Medical Image Analysis Assignment 4

Casper Bresdahl

1 Introduction

In this week's assignment we will look at the basics behind CNNs and we will look at how well a U-Net model can segment lung data. Then we will look at cross validation, before we end by looking at a skin lesion classifier and how we can use different metrics to evaluate its performance.

2 Basics of CNNs

Convolutional neural networks are a sub class of deep neural networks which mainly consists convolutional layers. A convolution uses a kernel which slides across the image and performs a weighted sum over all affected pixels in the image. Kernel sizes of 3 by 3 or 5 by 5 are the most common. The output of the convolution is a new image which is smaller in size than the original image. To overcome this, a number of different padding strategies can be used, the most common is to pad the original image with zeros such that the output of the convolution has the same size as the input image. Different kernels produces different outputs, for instance putting equal weight in each kernel index will blur the original image as the result of using such kernel is the average pixel value in a local neighbourhood. Defining a 3 by 3 kernel where the top row are ones, the middle are zeros and the last are negative ones will find horizontal lines where we have background below the edge.
Convolutions can find patterns in a local neighbourhood, but not in a global setting, thus max pooling layers are often used in CNNs to shrink the size of the image but to keep the important features found by the convolutions. Max pooling works by taking the max value in a local image patch, often a 2 by 2 patch. Performing a sequence of convolution(s) followed by a max pooling will result in the image shrinking and the convolutions in a sense can find features at bigger and bigger scale as the relative area of the image the kernel operates increases when the image shrinks. This makes CNNs scale invariant and allows for object detection regardless of the object being in the corner of the image, or if it fills the whole image.
Each convolutional in the network detects a certain feature, this could be diagonal lines or round object. As we at some point need to Linearly combine these responses to make a prediction based on the responses of the convolutions, we are not interested in for instance how 'not round' an object is. For this reason we use activation functions. These are non-linear functions like ReLU which removes negative responses but keeps positive responses. Another example is the sigmoid function which gives a probability. Using activation functions ensures

the 'not roundness' of an object does not skew the result of the network as we are only interested in if the object is round.

3 U-Net for image segmentation

In this section we will use *keras* along with the library *segmentation_models* to create a U-Net model and train it on our lung data. The model have been trained on 99 images, validated on 25 images, and tested on 123 images. For loss function *segmentation_models*'s *dice_loss* have been used along with an Adam optimizer initialized with learning rate 0.0001. Dice loss have been chosen as we are dealing with a segmentation task where it seems more intuitive to measure the loss based on the overlap between the prediction and ground truth rather than pixel wise accuracy (where the wrongly predicted pixels might drown in the number of correct predictions). An alternative could have been to use intersection over union. To measure the performance on the test set, we use f1 score which is almost the same as the DICE coefficient, expect we weight correct predictions higher. As there is no DICE metric in *keras* and *segmentation_models* f1 score has been used instead. In Figure 1 we see the training and validation loss.

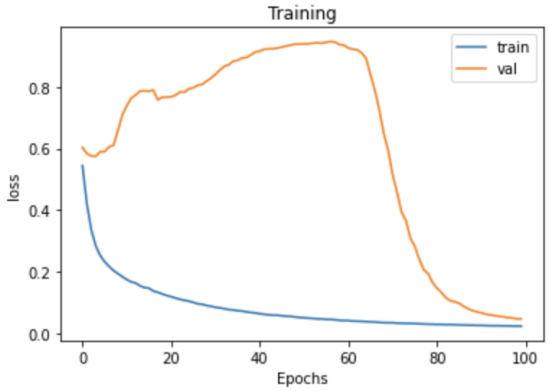
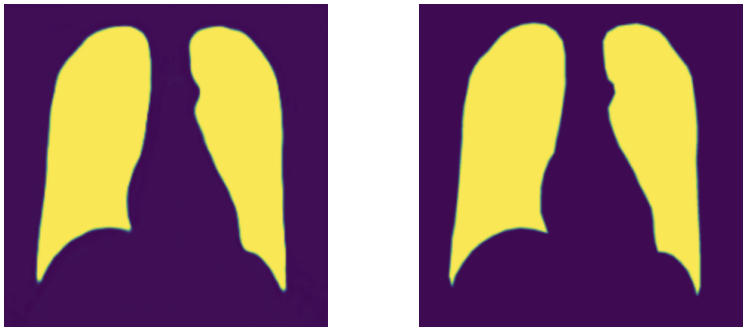


Fig. 1: Training and validation loss while training (measured as DICE loss).

The validation loss starts off by growing quite a lot before at around 65 epochs to drop very rapidly. This indicates that the model around this point finds one or more features in the images which are good to use for segmentation and turns the emphasis / weights for these features up quite quickly. The training loss seems to behave quite normally by dropping quite steadily.
For the **test set** we can report an average **f1 score** of **0.955**. This is quite high, and shows the model both generalizes and performs quite well.
In Figure 2 we see an example of an prediction along the ground truth for that prediction. The DICE coefficient for this prediction is 0.988. As we can see, and

the DICE coefficient indicates, the prediction is quite good, with only some of the edges being a little imprecise.



(a) Prediction. (b) Ground truth.

Fig. 2: Model prediction and ground truth. Their DICE coefficient is 0.988.

4 Cross validation

When training machine learning models it is important to split your data into three categories, training, validation and test. The training set is the data the model is trained on, and models will tend to overfit this data. Because of this, we use a validation set to validate that the model generalizes to unknown data, that is, we do not train on the validation data, we only evaluate performance on it. The test set is only used at the very end when the final model evaluation is performed. If we have loads of data, we can simply split our data into these three categories and keep them static. But if we only have a small dataset we can use the cross validation technique. Here our data is split into a training set and a test set. The test set is as always only used at the very end. However, the training set is then further divided into 'splits'. A series of 'folds' are then performed. At each fold, one split is kept as a validation set and the rest is used as training data. As we complete a fold we move to the next where a new split is used as validation set and the rest is used as training data. Using this approach, each split will eventually be held out and used as validation data. At the end, the model performance is evaluated on the test set. Using this approach we utilize our data more efficiently and is to prefer when we have very little data to work with. However, as we still train on our validation data we do not get as accurate an indication of how good the model generalizes on unknown data. Also more care needs to be taken when using cross validation as we need to ensure each fold has the same proportion of observations with a certain property. When creating static data sets we only need to ensure this for the three data sets, but for cross validation we need to ensure this for each of the k folds plus the test set. An illustration of cross validation can be seen in Figure 3.

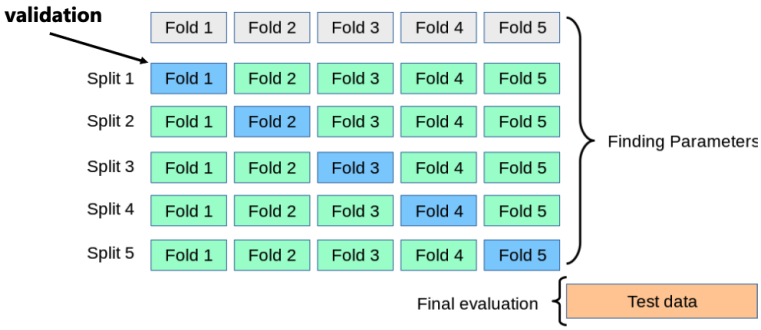


Fig. 3: Illustration of cross validation. Each of the blue splits are used at turn as validation sets, whereas the green splits are used training sets. Illustration taken from slides.

5 Skin lesion classifier

For the skin lesion competition I developed a model in collaboration with Marcus Hansen, Sarah Hansen and Ulrik Larsen. This model consists of six blocks of 3 by 3 convolutions with a ReLU activation function followed by a batch normalization followed by a 2 by 2 max pooling. For the convolutions padding with zeros have been used to keep the result after convolutions the same as the input size. After these six blocks, three blocks of a fully connected layer with a ReLU activation function followed by a batch normalization is used. The first fully connected layer has size 32, the second 16 and the last has 8. Lastly we have a fully connected layer of size 1 with a sigmoid activation function to get the class probability. This model has been trained with binary cross entropy as the loss function and an Adam optimizer with initial learning rate 0.00001. For the competition this model was trained for 1000 epochs using random affine transformations on the training data. The **final training accuracy** was around **0.90**, the **final validation accuracy** was around **0.80** and the **final test accuracy** was around **0.70**.
To evaluate this classification model I have been retraining the model, however, due to limitations in hardware I have only been able to train for 100 epochs. To compensate for this, I have used a learning rate of 0.0001 (a factor 10 larger) instead. The metrics I have chosen to use to evaluate the model are *accuracy*, *area under the ROC curve* and *area under the precision / recall curve*. Accuracy simply tells us how many correctly classified images we got, i.e. true positives plus true negatives. Area under the ROC curve is given by measuring the area under the curve of a ROC curve. A ROC curve is made by plotting the sensitivity vs 1-specificity and tells us about the trade-off between the true positive rate and the false positive rate, i.e. the percentage of true positives we expect to find if we allow for some false positives. The area under the precision / recall curve is given by measuring the area under a precision / recall curve. Precision / recall curves measures how good the model is at predicting the positive class

vs the ratio of true positives and false positives. Precision / recall curves can be relevant when we have imbalanced data, as the curve describes how often we predict true positive without introducing false negative, that is, we disregard the true negative 'accuracy'.

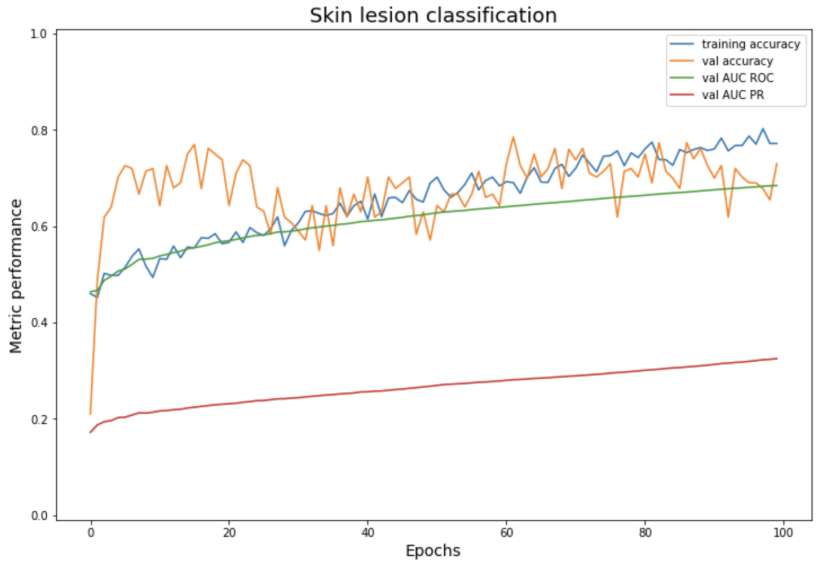


Fig. 4: Performance of the retrained skin lesion classifier.

In Figure 4 we see the training and validation accuracy along the area under the ROC curve and area under the precision / recall curve. We note these results are not as good as the results of the original model based on the validation accuracy. However, an accuracy of about 0.7 on unknown data is not ideal, but the model has learnt something. We note both the validation and training accuracy are still increasing and so, more epochs would probably have improved the model. When looking at area under the curve for ROC, we note it is steadily improving, which means we are getting better and better at predicting true positives without having to include more false positives. Lastly we note the area under the curve for precision / recall is also steadily increasing, which means we are getting better at predicting true positives without predicting false negatives.

6 Conclusion

In conclusion we have seen CNNs are build by using convolutions and we have defined cross validation and when it can be useful. We have also seen a U-Net model predict quite good segmentations for lung data. And we have evaluated the performance of our retrained skin lesion classifier, by measuring accuracy, area under the ROC curve and area under the precision / recall curve.

7 Code collaboration

The code for this assignment has been developed in collaboration with Marcus Hansen, Sarah Hansen and Ulrik Larsen.