Ouadahi Fares
Cama Hector

LAB5 - Report

Task 1) : K-fold cross validation

The goal of this task is to evaluate the model on different training and test sets using K-fold cross validation. After implementing the K-fold function, we perform the segmentation of brain tumors from MRI images.
The results obtained are consistent through the three folds.
If the performance in one fold was lower, we could think that it is a class-imbalanced problem and so that we could use other sampling methods such as stratification sampling. We use 150 epochs and the final accuracy is always about 0.85.

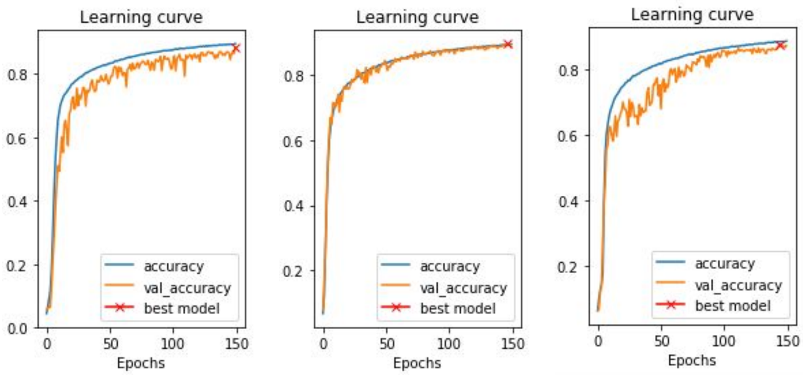

Figure 1: Training curves for the K-fold cross validation. Each curve corresponds to a different training and test set.

Task 2) : Introducing weight maps

In this task, we want to implement weight maps to improve the segmentation. So we associate different weights to the mask boundaries. To do that, we use erosion and dilation methods to keep only the boundaries as the weight map. Then we implement the weighted loss function. The results for 100 epochs are the following :

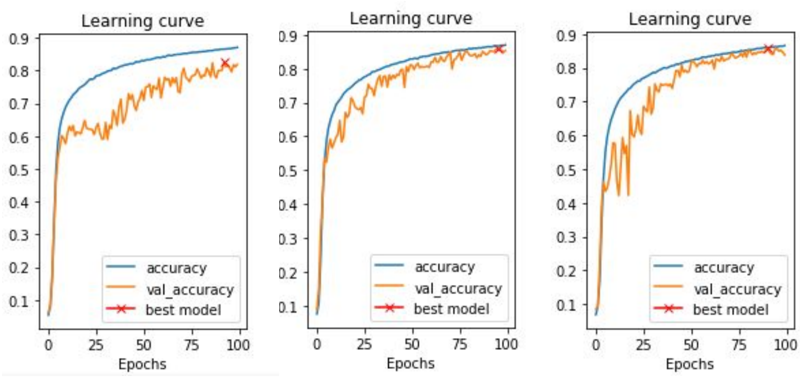

Figure 2: Training curves for the K-fold cross validation with weight maps. Each curve corresponds to a different training and test set.

For a smaller number of epochs, the accuracy is the same as the previous task if not better with a weight map. It shows that the weight map makes the segmentation easier.

Task 3 : Adding autocontext

The last step of FCN improvement is adding autocontext. It means using the segmentation result from a step s as an input to the step s+1.

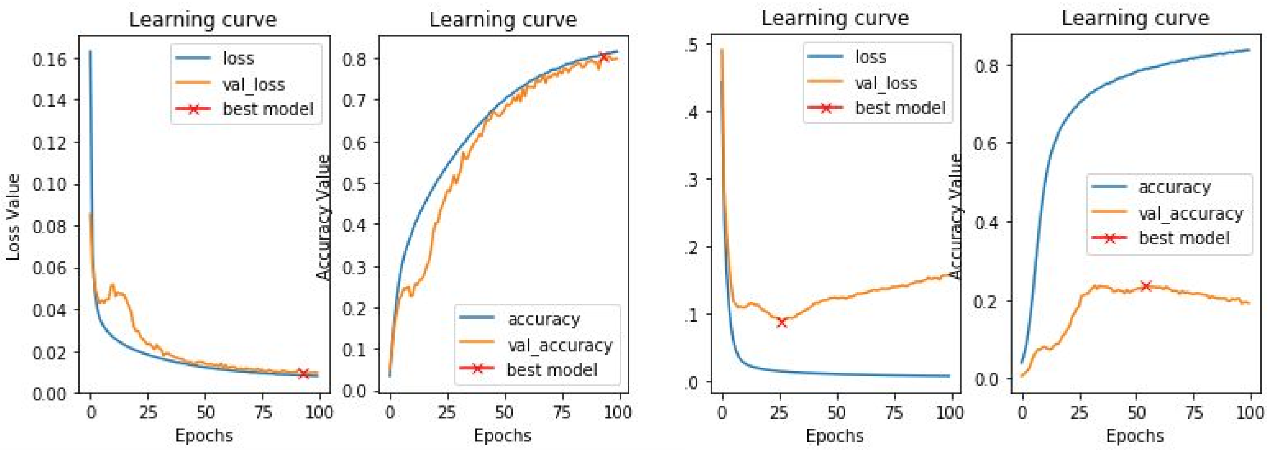

Figure 3 : Loss and accuracy curves for step 0 and step 1 for one fold with the autocontext

As we can see on figure 1, the first step with the initialization seems to give good results for the FCN output (around 0.8 accuracy). However, we get an overfit by adding the autocontext layer into the model.

To sum up the results, the weighted U-Net seems to be the best model of the three. However, it could be possible to improve the performances of the autocontext model, maybe by increasing the dropout rate (0.2 in our case), or by adding weight maps into it.