Cristian Marti c.marti@methinks.es, Carlos Crespo c.crespo@methinks.es
Methinks Software S.L. Barcelona, Spain.

Abstract

The following document describes work done in order to complete the Promise 12 challenge. The challenge consists in the automatic segmentation of the prostate in transversal T2-weighted MR images. The data includes both patients with benign disease (e.g. benign prostatic hyperplasia) and prostate cancer.
We use Deep Learning in general, and some specific techniques in particular, in order to tackle the problem. The details are described next.

Data preprocessing.

Resampling.

We have converted volumes with a spacing for x and y axes less than 0.4 mm, into 4 different volumes with the following characteristics:
Volume 1: even indexes in x, even indexes in y, all indexes in z.
Volume 2: odd indexes in x, odd indexes in y, all indexes in z.
Volume 3: odd indexes in x, even indexes in y, all indexes in z.
Volume 4: even indexes in x, odd indexes in y, all indexes in z.
In this way, the spacing for these new images is doubled with respect the original and we end up with all volumes having a similar spacing 0.46 <= spacing <= 0.78
Apart from this, we have whitened each volume, subtracting the mean of the whole volume from the value of each voxel and dividing it by the standard deviation of the whole volume.
The four segmentations produced for each volume that has gone through this process are averaged, and interpolated to create a unique prediction from them.

Distance mask.

In order to take advantage from the fact that in all train data set and test data set samples the prostate is located in a similar relative position with respect the whole volume sample, we have added an additional channel to the records to be used as input for the network, in which each voxel represents the distance from it to the spacial point (volume_size_x / 2, volume_size_y / 2, volume_size_z / 4) in millimeters, which is the rough location of the prostate in the volumes. Our intention is to allow the network to learn that there is a higher probability of finding prostate in regions in which the value in the mask is small.

Data augmentation.

The only kind of data augmentation applied is an x axis flip performed with a 50% of chance for each sample.

Training input.

Due to memory constraints, we are not able to process a whole MRI volume in a training step, for this reason we process sub-volumes extracted from the original image. The sub-volumes are extracted in such a way that 80% of the time will be centered on a random voxel within the prostate, and the other 20% on a random voxel outside the prostate.

Network Structure.

For our neural network we have selected a fully convolutional neural network with a Unet like architecture. We have a downsampling path and an upsampling path with shortcut connections between volumes having roughly the same dimensions.
We also have used ResNet blocks as our basic construction block in our network.

Taking into account volume anisotropy.

As the spacing for z dimension is between 2 and 4 for all the samples, the information in this dimension is scarce compared with x and y which have significant lower spacing values. For this reason, we are careful to not use strided convolutions (in the downsampling stage of the network) in this specific dimension, and in general use convolutions with kernel size equals to 1 in z, except for two convolutions with kernel 3.

Padding valid.

In order to avoid blocks to be affected by 0 padding added during it way through the network, we only use padding valid in our convolutions, this is: not padding is added at any point in the network. This makes the final prediction smaller when compared to the original sub-volume, due to the padding loss, which are the voxels lost due to convolutions effect. Although this makes the training process slower, it guarantees that predicted voxels values are the same as if we would have been able to process the whole MRI volume at once.

Downsampling path.

During the downsampling path, we apply convolutions with stride 2 for dimensions x and y, in order to roughly halve the size of the volume in those dimensions. In z, we use stride 1 to preserve the size in that dimension taking into account the anisotropy in the volumes.
During the downsampling path, this procedure is repeated 3 times, obtaining volumes that are roughly ½, ¼ and ⅛ of the original one in x and y dimensions.

Upsampling path.

During the upsampling path, we apply transposed convolutions with stride 2 for dimensions x and y, in order to roughly double the size of the volume in those dimensions. In z, we use stride 1 to preserve the size in that dimension taking into account the anisotropy in the volumes.
During the upsampling path, this procedure is repeated 3 times, obtaining volumes that are roughly ¼, ½ and the same size of the original one in x and y dimensions.

Deep supervision.

We take volumes produced during the upsampling path to create 2 auxiliary predictions of roughly ¼ and ½ of the original input which is then upsampled with transpose convolutions to roughly the same size of the original input, the last volume of this path with roughly the same size of the original input is used to generate the main prediction. All the predictions are used for the loss metric calculation and the inference process.

Average feature maps.

We add additional feature maps to the sub-volumes being processed in different points of its travel through the network. Each voxel of these added feature maps, corresponds to the activation average of its voxel neighborhood for a specific feature map. The reason to do this, is to help the convolutional filters to see what is happening around each voxel in terms of activations.

Regularization.

We use dropout (85% of keep probability on each hidden layer) and batch normalization as methods for regularization.

Loss metric.

For the loss metric, we calculate the probability maps of all the output predictions applying softmax to each one, then we calculate the cross entropy with respect the ground truth for each of them. We perform a weighted sum of the cross entropy of the two auxiliary predictions and the main one.

Pseudo-labeling.

We have used pseudo-labeling as a Semi-Supervised Learning Method to achieve Entropy Regularization. After a complete training session, we create labels for the test set using the resulting model, and then use the test set with these new labels as part of the training set to train a new and final model.

Evaluation.

Implementation details.

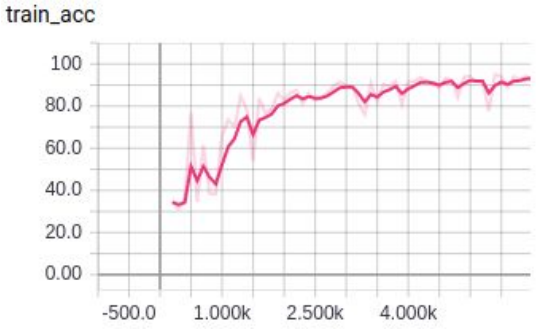
We have implemented our model using TensorFlow. We have used 3 NVIDIA Quadro M6000 GPUs. Our sub-volume size is 240x240x14 and our batch size is 15. The training process is executed. We have used Adadelta optimizer.

Inference process.

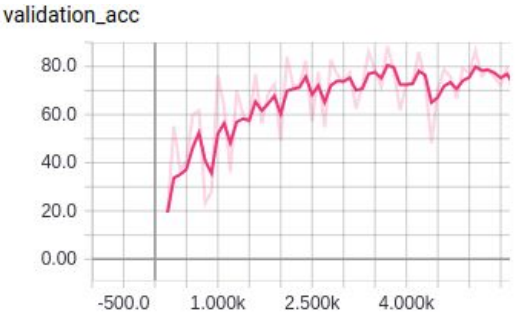
In order to process a whole MRI volume, we transform it into contiguous sub-volumes, generate predictions for each of them and then assemble them together in order to obtain the main prediction. The stride for each dimension used when extracting the blocks is not exactly the block size in that dimension, but the block size minus the padding loss of the network, this overlapping between input blocks allow to end up with prediction blocks that can recreate the whole volume without gaps or overlapping.

Results.

We have removed 5 samples from the training set, to use them as validation set. The selected cases were 00, 01, 03, 13 and 49.
The only metric we have used is
In our training set we achieved a dice score of 0.95.
In our validation set we achieved a dice score of 0.81.



Training accuracy with respect training steps.



Validation accuracy with respect training steps.