Brain Tumor Segmentation

Egor Krivov Egor Kuznetsov

I. Introduction

Segmentation and the subsequent quantitative assessment of lesions in medical images provide valuable information for the analysis of neuropathologies and are important for planning of treatment strategies, monitoring of disease progression and prediction of patient outcome. For a better understanding of the pathophysiology of diseases, quantitative imaging can reveal clues about the disease characteristics and effects on particular anatomical structures. The quantitative analysis of lesions requires accurate lesion segmentation in multi-modal, three-dimensional images which is a challenging task for a number of reasons. The heterogeneous appearance of lesions including the large variability in location, size, shape and frequency make it difficult to devise effective segmentation rules.

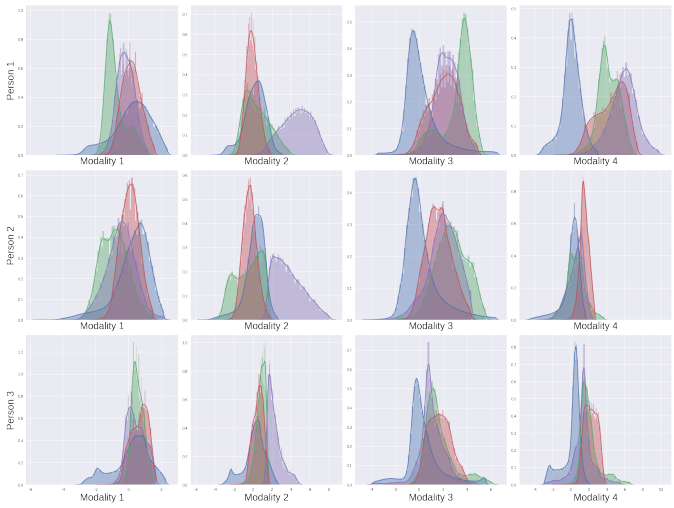

Fig. 1. Intensity distribution for different cancer tissues (the enhancing tumor (green), the tumor core (violet), the whole tumor (red) and healthy tissue (blue))

Fig. 1 illustrates some of the challenges that arise when devising a computational approach for the task of automatic lesion segmentation. Image intensity profiles largely overlap with non-affected, healthy parts of the brain or lesions which are not in the focus of interest.

In this work, we try to tackle the aforementioned problem in the context of "Multimodal Brain Tumor Segmentation Challenge 2017" (BraTS 2017) (http://braintumorsegmentation.org/).

II. Dataset

The database of BraTS 2017 includes multi-modal 3D MRI scans for 285 patients. These multi-modal scans describe a) native (T1) and b) post-contrast T1-weighted (T1Gd), c) T2-weighted (T2), and d) T2 Fluid Attenuated Inversion Recovery (FLAIR) volumes, and were acquired with different clinical protocols and various scanners from multiple institutions. The size of each modality is $240 \times 240 \times 150$. To decrease the size of the scans we clip edges with unimportant information and leave only part with a brain.

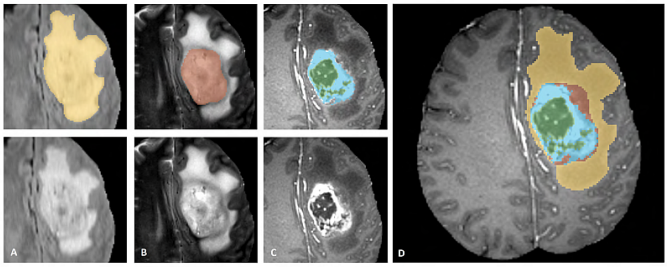

Fig. 2. Glioma sub-regions. Shown are image patches with the tumor sub-regions that are annotated in the different modalities (top left) and the final labels for the whole dataset (right). The image patches show from left to right: the whole tumor (yellow, Fig.A), the tumor core (red, Fig.B), the enhancing tumor structures (light blue, Fig. C).

Our goal is to provide a segmentation of each MRI scan into 3 binary masks (see Fig. 2): the whole tumor, the tumor core, the enhancing tumor structures.

III. Deep medic architecture

First of all, we try to investigate the existing approaches to this problem. One of the best architecture, which is used to tackle this problem, is described in [2]. This architecture is called DeepMedic and depicted in Fig. 3.

DeepMedic is a multi-scale 3D CNN with two convolutional pathways. These pathways process the input at different scales to achieve a large receptive field for the final classification while keeping the computational cost low. The inputs of the two pathways are centered at the same image location, but the second segment covers a large area. Instead of down-sampling patch before feeding it into the net, we add Average pooling layer. Also at the end of the second pathway, we replace the up-sampling layer with transposed convolutional layer. The operations within each layer block are applied in the order: Batch-Normalization, non-linearity and convolution. Number of filters and their size depicted as $(Number \times Size)$.

In this architecture the residual connections between the outputs of every two layers, except for the first two of each pathway, are used. In this case, a residual connection after layer l performs the element-wise addition $\oplus$ between corresponding channels of the output of layer l and the input

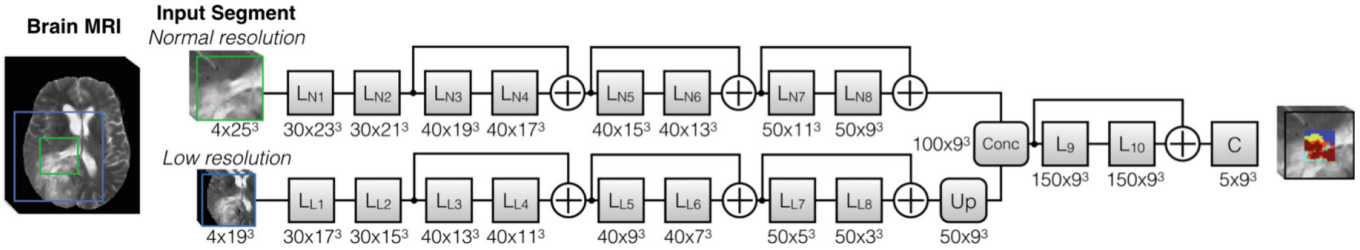

Fig. 3. DeepMedic architecture

to the previous layer $l-1$. More formally:

$$in^m_{l+1} = \begin{cases} out^m_l \oplus \hat{in}^m_{l-1}, & \text{if } m \leq M^{l-1}, \\ out^m_l, & \text{otherwise,} \end{cases}$$

where l denotes any layer after which a residual connection is added, the super-script m is the m-th channel and M^l is the number of feature maps in the l-th layer. $\hat{in}_l$ is the input of the previous layer after clipping in the (x, y, z) dimensions in order to match the dimensions.

Training is performed with Adam optimization method. Batch size equals 100. Number of batches per epoch equals 40. Learning rate is chosen manually: 0.1 for the first 10 epochs, 0.01 for the next 14 epochs and 0.001 for the next 5 epochs.

IV. EE-NET

We also try to implement our own network architecture, which is highly inspired by ResNet-50 architecture, especially bottleneck blocks [1]. It is a simple sequential architecture consisting of 4 bottleneck blocks, see Fig. 5. We call our architecture EE-net, since E-net name was already taken [3]. We tried to implement one of the latest and most effective network architectures. To our knowledge, in the field of image processing, only inception architecture is a reasonable match for ResNet architecture as well as a combination of inception ideas with ResNet [4].

Each bottleneck block, visualized in Fig. 4 consists of two pathways: processing and optional transformation. Processing pathways consists of three modules: compression, which is aimed at feature map compression with 1^3 convolutions, inner processing, which performs spatial convolution with 3^3 kernels and decompression, which increases the number of feature maps. Optional transformation is performed only if the number of output layers is different from the number of input layers. It is performed with convolution, followed by batch normalization. Finally, we sum up two pathways voxelwise and apply final nonlinearity.

Training is performed with Nesterov momentum. Leaning rate is chosen and changed manually to get better convergence. Learning rate sequence is $0.1, 0.05, 0.01, 0.05, 0.01, 0.003$. Batch size was 128.

V. EVALUATION

To measure performance of our models we randomly split the dataset into training, validation and test sets: 222, 6 and 57 scans respectively. We choose this split only once and use it for both models.

As a metric of quality we use dice score, which is typical for medical image segmentation and is used in the BraTS 2017 competition. Dice score for one patient and one segmentation mask is:

$$DSC(p, g) = \frac{2\sum_i p_i g_i}{\sum_i p_i + \sum_i g_i},$$

where $p_i \in \{0, 1\}$ is model's prediction for i-s voxel and $g_i \in \{0, 1\}$ is a ground truth value for the same voxel.

Since we have 3 segmentation problems for each scan: the whole tumor, the tumor core, the enhancing tumor structures, we will have 3 dice scores for each patient. We average them across patients to get final metric for each of 3 tumor types. This metric requires discrete predictions, meaning that we have to choose three thresholding values. We choose them by maximizing each of 3 average dice scores on validation set.

To train our model we use binary cross entropy (logloss) between actual bitmaps and predicted. Therefore, all types of cancer are equally important for segmentation. We have also tried using continuous version of dice score as a loss function, to optimize it directly and achieve better performance. But results were worse than with binary cross entropy.

$$DSC_{cont}(p, g) = \frac{\varepsilon + \sum_i p_i g_i}{\varepsilon + \sum_i p_i + \sum_i g_i},$$

where $p_i \in [0, 1]$ is model's probability prediction for i-s voxel and $g_i \in \{0, 1\}$ is a ground truth value for the same voxel.

Training was performed with pytorch on titan X and GTX 980 Ti and took less than an hour to train for each model. Example of segmentation for EE-net is depicted in Fig. 6

VI. CONCLUSION

In this work we have tried to approach the problem of medical image segmentation by reproducing results of the state-of-the-art solution and implementing our own solution. Results are presented in Table I. To compare our results with DeepMedic we provide results of the original paper, although it should be noted that our dataset is different. Our dataset have about 75 scans in common with their dataset (Our whole dataset have 285 scans). Based on the final results we note that in our implementation, our architecture performs better than DeepMedic by a significant margin.

Further improvements of our model would first of all include dilated convolutions, to use context information.

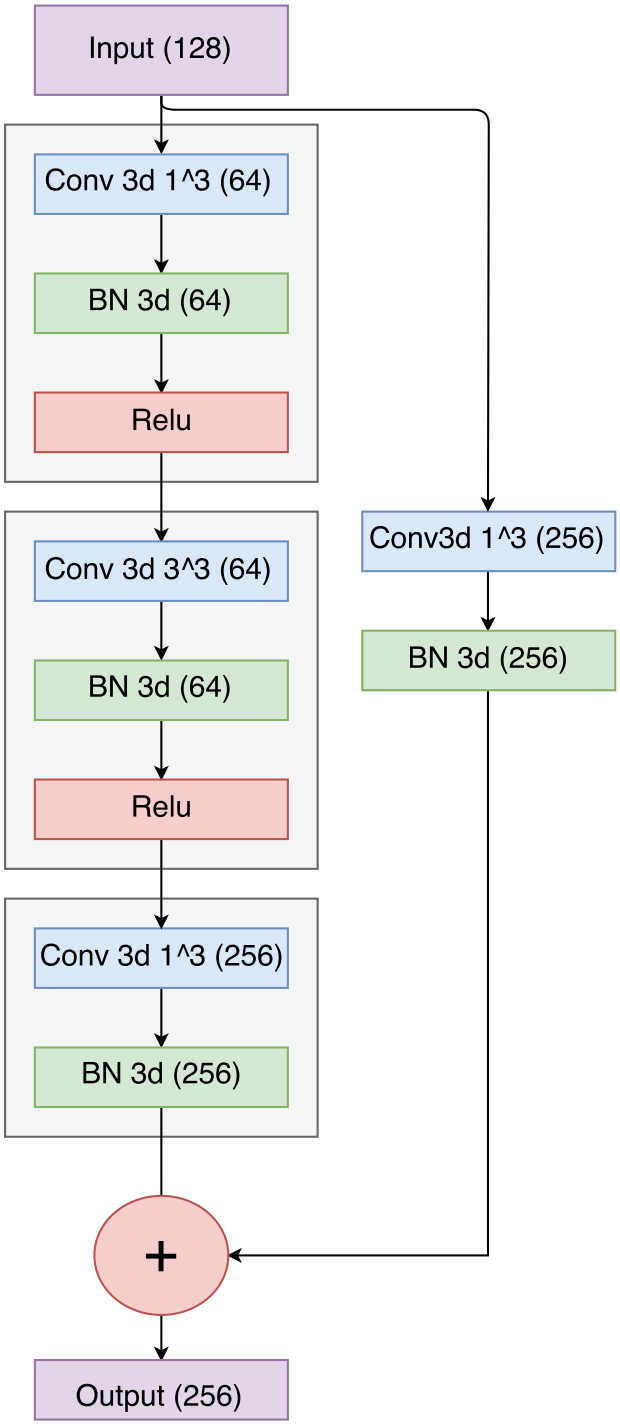

Fig. 4. Bottleneck (128, 64, 256) block

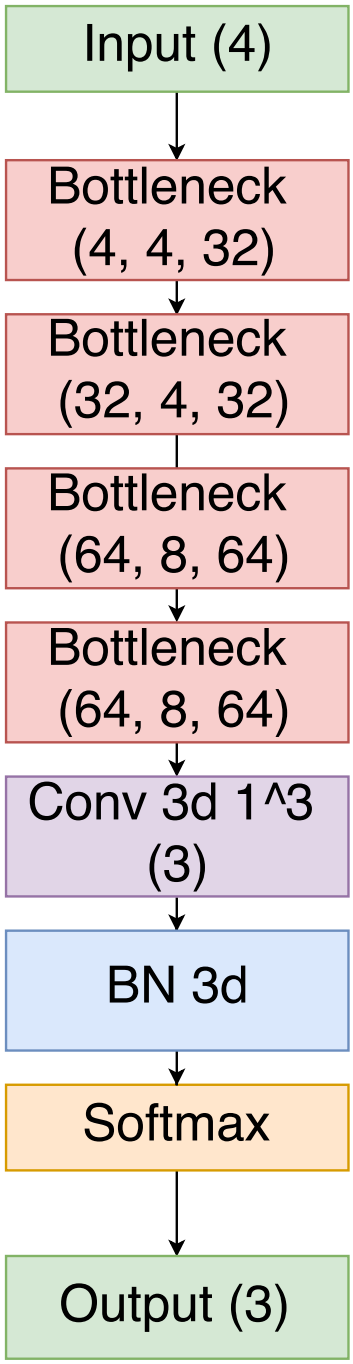

Fig. 5. EE-Net architecture

TABLE I
RESULTS

	DSC Whole	DSC Core	DSC Enh.
DeepMedic from the paper*	0.896	0.763	0.724
DeepMedic (our impl.)	0.771	0.558	0.467
EE-net	0.816	0.693	0.669

*Results of DeepMedic were taken from the paper [2] and correspond to partially different dataset

VII. CONTRIBUTIONS

Work in our team is based on equality and mutual assistance. The key rule is that both members of team must know and understand each part of the project.

Egor Kuznetsov studied the article [2] and implemented the DeepMedic architecture. He carried out experiments with this network and tuned parameters. Also, he implemented Dice loss and tried to play with it. Egor made contribution to the implementation of the batch iterator, which chooses random

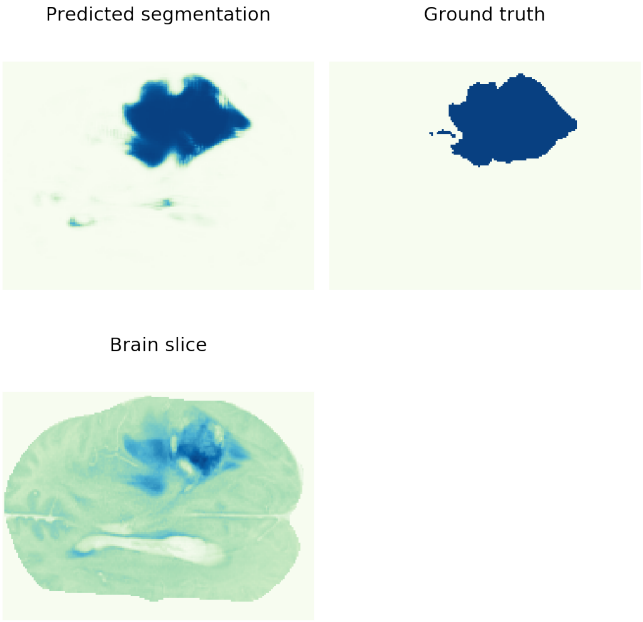

Fig. 6. Segmentation example

patches from 3D scans.

Egor Krivov have downloaded the data and performed preprocessing. He also wrote part of batch iterators and developed EE-net and trained it.

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[2] Konstantinos Kamnitsas, Enzo Ferrante, Sarah Parisot, Christian Ledig, Aditya Nori, Antonio Criminisi, Daniel Rueckert, and Ben Glocker. Deepmedic on brain tumor segmentation. *Athens, Greece Proc. BRATS-MICCAI*, 2016.

[3] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016.

[4] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.