Acta Univ. Sapientiae Informatica **14,** 1 (2022) 49–74

DOI: 10.2478/ausi-2022-0004

U-Net architecture variants for brain tumor segmentation of histogram corrected images

Szidónia LEFKOVITS
George Emil Palade University of Medicine, Pharmacy, Science, and Technology Târgu Mureş, Department of Electrical Engineering and Information Technology
email: szidonia.lefkovits@umfst.ro

László LEFKOVITS
Sapientia Hungarian University of Transylvania, Cluj-Napoca Department of Electrical Engineering, Târgu Mureş
email: lefkolaci@ms.sapientia.ro

Abstract. In this paper we propose to create an end-to-end brain tumor segmentation system that applies three variants of the well-known U-Net convolutional neural networks. In our results we obtain and analyse the detection performances of U-Net, VGG16-UNet and ResNet-UNet on the BraTS2020 training dataset. Further, we inspect the behavior of the ensemble model obtained as the weighted response of the three CNN models. We introduce essential preprocessing and post-processing steps so as to improve the detection performances. The original images were corrected and the different intensity ranges were transformed into the 8-bit grayscale domain to uniformize the tissue intensities, while preserving the original histogram shapes. For post-processing we apply region connectedness onto the whole tumor and conversion of background pixels into necrosis inside the whole tumor. As a result, we present the Dice scores of our system obtained for WT (whole tumor), TC (tumor core) and ET (enhanced tumor) on the BraTS2020 training dataset.

Computing Classification System 1998: I.4.6
Mathematics Subject Classification 2010: 68T07
Key words and phrases: brain tumor segmentation, histogram uniformization, U-Net, VGG16-UNet, ResNet50-UNet, BraTS2020

1 Introduction

The automatic processing of medical images is one of the fastest-growing domains in the field of computer assisted medical diagnosis and surgery. The automation systems concentrate on the use of various artificial intelligence methods relying mostly on supervised learning. Generally, the computing algorithms have the role of identifying different malformations in the healthy tissues. Their role is to localize them by applying a detection algorithm or to segment the image, determining the class of each pixel. These algorithms are based on an enormous amount of data on which the algorithms have to be trained in order to obtain a capability for generalization on unseen images. In this paper we concentrate on medical image segmentation, namely, on brain tumor segmentation from MRI (Magnetic Resonance Image) data. The tumors in the brain can develop into two types: LGG (Low-Grade Glioma) and HGG (High-Grade Glioma). According to the WHO, tumors can be categorized into 4 subtypes: Grade I and II are LGG and III and IV are HGG. Grade I is a benign and slow-growing tumor, while grade II presents increased hypercellulality and can develop into a HGG tumor. Grade III and IV are much more severe variants. Grade III shows a high rate of hypercellularity and a high rate of mitosis, while Grade IV is an advanced form of Grade III in which necrosis and vascular proliferation occur [10]. Early diagnosis is essential in the treatment of the disease and facilitates timely initiation of treatment or other necessary medical interventions. It is also essential to note that computerized detection is a rapid process providing a non-contact diagnosis tool for physicians. The worldwide spread of MRI equipment and the growing number of MRI records acquired should make the general screening of the population possible, thereby detecting cases in an early stage.

The bottlenecks of an automatic brain tumor segmentation are the great variety of positions, dimensions, aspects and textures of the tumor; the deformation of the normal tissues; the wide variety of MRI equipment used without a given standardized protocol; the varying resolutions and pixel value inhomogeneity produced by different MRI machines.

MRI image processing methods can be classified into two large categories: traditional image processing methods and CNN methods. The pre-deep learning era concentrates on traditional pattern recognition methods, such as thresholding, region-based and pixel classification methods or model-based techniques. Thresholding techniques include global thresholding, local thresholding or Gaussian distribution-based thresholding. Region-based methods search for common features of a given region such as homogeneity, similarity, region grow-

ing and watershed segmentation. Pixel classification methods and model-based methods are combined in various ways. According to the literature, this category comprises clustering methods, fuzzy c-means clustering, Markov Random Fields, Artificial Neural Networks, Self-organizing Maps, Random Forest [18], Boosting, Geometric Deformable Models, Level Set Methods, Charged Fluid Model and so on. Paper [12] presents a review of the above-mentioned methods.

The organization of the BraTS – Brain Tumor Segmentation World Grand Challenges represents a major leap in the development of brain tumor segmentation [29, 20, 4, 2]. This Challenge has been organized since 2012 by the MICCAI Society at the eponymous conference. From then on, the organizers, including dozens of contributors, have been gathering MRI images from hospitals and research centers. Until 2015, all methods were based on pre-deep learning models. In 2015-2016, the new deep learning technique expanded into the field of medical imaging as well. In [21] Pereira et al. propose CNNs exploring small kernels of 3×3 pixels to obtain deeper network architecture with a lower number of weights than larger kernels. In [28] the authors propose a two-step segmentation for brain tumors: first the authors train a Fully Convolutional Neural Network (FCN) and then train Conditional Random Fields as Neural Networks.

Kamnitsas et al. [16] describe a 3D 11-layered deep neural network in which they process adjacent image patches in a single pass through the networks to handle class imbalance. In order to handle local and global features, they propose a dual-pathway multi-resolution network. Xue et al. [27] introduces a novel Generative Adversarial Network (GAN) for medical image segmentation. In the GAN, the segmentor is an FCN producing segmentation label maps, while the adversarial critic is a multi-scale CNN with L1 loss. Ding et al. [11] describes the Deep Residual Dilate Network with Middle Supervision (RDM-Net) that combines the residual network with delated convolution for multi-modal brain tumor segmentation. Chen et al. [9] proposes a dual-force training scheme for the extended version of DeepMedic, leading to more accurate segmentation in the Multi-Level DeepMedic thus obtained. Wu et al. in article [26] introduces the Multi-features Refinement and Aggregation (MRA) that utilizes hierarchical features by combining features from different levels into an aggregation module passing the joined response through a second CNN. Isensee et al. [15] developed a network that configures itself automatically considering 2D and 3D U-Net and U-Net Cascade. It computes the number of downsampling/upsampling form the resolution of the data, the class ratio and the image size after cropping the background. In addition, it determines the

patch size, the batch size, the optimization function and the number of images to be augmented in order to fit into the GPU memory. This architecture has participated at the Medical Segmentation Decathlon Challenge [31] and has achieved one of the highest mean Dice scores in all segmentation challenges.

All referenced articles written after 2015 rely on the BraTS dataset collected between 2012-2020. The most up-to-date publicly available dataset was published in the TCGA-GBM [5] and TCGA-LGG collections [6].

The main contribution of this paper consists of two major steps: we apply the histogram uniformization described below as preprocessing for the image dataset and adapt the well-known U-Net architecture for the purpose of brain tumor segmentation. In addition to applying only the Vanilla-UNet, we also adapt and train VGG16-UNet, the ResNet50-UNet versions of the U-Net. We also create an ensemble model combining the probability responses of the 3 mentioned models.

The paper is organized as follows: following this first introductory section and a short literature review, our system is described, detailing the methods and models applied and going through every component of the segmentation pipeline. Finally, we present the results obtained and suggest a number of important improvements in the conclusion.

2 Materials and methods

Our proposed automated segmentation system (Figure 1) is based on different U-shaped deep convolutional neural networks. For semantic segmentation, the traditional convolutional neural networks are transformed into fully convolutional neural networks (FCNs). Our system is trained on the BraTS 2017-2020 database.

The original 3D images are preprocessed at the beginning to provide uniform tissue intensities on all the images. In the training process, the data is divided into training, validation and test sets. The training stops when the Dice score does not increase any more. After obtaining the segmentations of the 3D brains, the connectedness of the tumors is verified. Finally, our results are evaluated using the well-known measures of accuracy on the test set.

2.1 Database

The image database used in this study is a publicly available database provided by the organization committee of BraTS (Brain Tumor Segmentation World Grand Challenge). The database consists of 369 cases of 293 HGG images

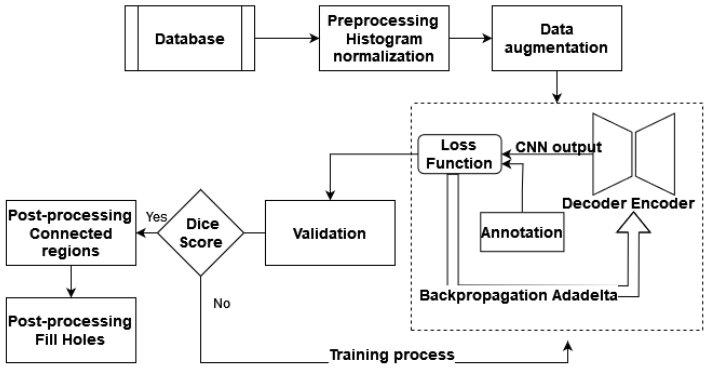
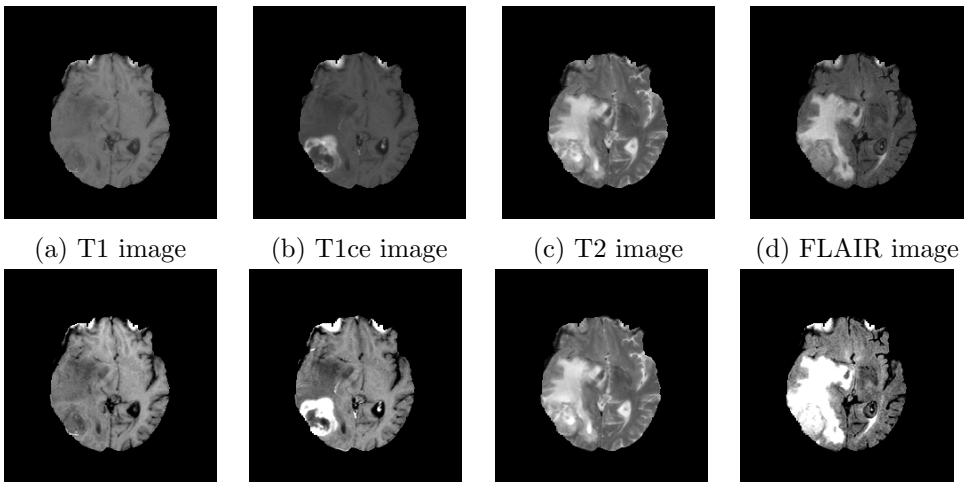


Figure 1: The proposed automatic segmentation pipeline



(a) T1 image (b) T1ce image (c) T2 image (d) FLAIR image

(e) corrected T1 (f) corrected T1ce (g) corrected T2 (h) corrected FLAIR

Figure 2: Histogram uniformization of the 4 modalities: 1. row: original images, 2. row: histogram unified images

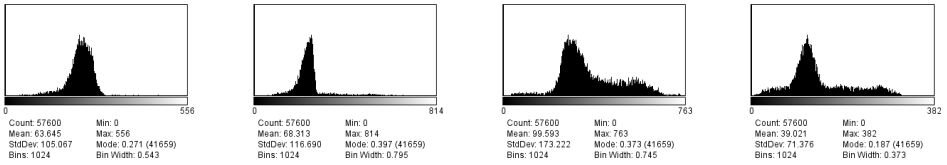
and 76 LGG images. The images are of course in 3D with a resolution of $240 \times 240 \times 155$ pixels. 1 voxel corresponds to $1\ mm^3$. A human brain is about $1500\ cm^3$, represented by 1.5 million voxels. The image format is NifTI, with every image being multi-modal containing 4 modalities: T1 image (longitudinal relaxation time), T1ce image (T1ce-weigthed with contrast agent), T2 image (transverse relaxation time) and FLAIR image (T2-Fluid Attenuated Inversion Recovery) Figure 2-row1. All these images are acquired via the MRI machine with different hardware setups. Every image was annotated by experts and represents the corresponding ground truth.

The image classes are: class0 is the background and the healthy tissue; class1-necrotic tumor core (NEC, joined with the non-enhancing tumor-NET); class2-peritumoral edema (ED), class4-enhancing tumor (ET). Class3 (label3) is missing; it was initially the non-enhancing tumor (NET).

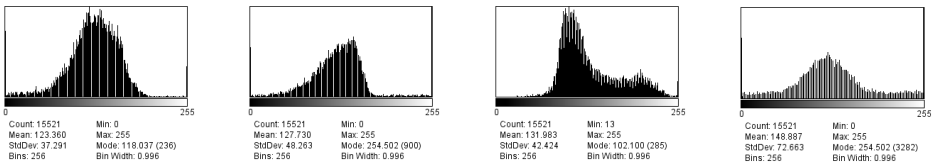
2.2 Preprocessing

The images in the original database are preprocessed applying the following steps: skull-stripping, they have the same resolution regardless of the MRI equipment and every image sets of the 4 modalities (T1, T1ce, T2, FLAIR) is co-registered to the same T1 template.

Intensity uniformization [13, 17] is an indispensable step in MRI image preprocessing. The differences in the setup and characteristics of the various equipment used to acquire images produce different intensity values for the same tissue. There is a wide range of intensity intervals that differs greatly from image to image. The same intensity grayscale value $I(x, y, z)$ can be the value of any of the possible tissue types (healthy, edema, enhanced or even necrosis) on different images. It is obvious that without obtaining the same intensity for the same tissue on different images the supervised learning cannot generalize and the segmentation results, without the preprocessing step, will be of low quality. The aim of the proposed histogram uniformization is to obtain the same intensity values for the same tissue over all images. Our uniformization method reduces the tissue intensity variance. We have chosen to transform the intensities between fist-quartile (Q_1) and third-quartile (Q_3) into a fixed experimentally defined interval $[int_1, int_2]$, because the intensity values in this range $[Q_1, Q_3]$ contain the major part of the image information. The interval $[int_1, int_2]$ is considered in the middle of the 8-bit grayscale range $[0, 255]$ (eq. 1). This histogram uniformization is a truncated linear transformation in which the values are limited into $[0, 255]$ (eq. 2). Every voxel of the



(a) T1 original histogram (b) T1ce original histogram (c) T2 original histogram (d) FLAIR original histogram



(e) T1 unified histogram (f) T1ce unified histogram (g) T2 unified histogram (h) FLAIR unified histogram

Figure 3: Original and corrected histograms of images in Figure 2

original image is transformed into the new 8-bit grayscale value by applying equations 1 and 2.

$$f(I(x,y,z)) = \frac{int_2 - int_1}{Q_3 - Q_1} \cdot I(x,y,z) + \frac{int_1 \cdot Q_3 - int_2 \cdot Q_1}{Q_3 - Q_1}, \tag{1}$$

In this formula some values may be exceed the $[0, 255]$ interval, so the $f(I(x,y,z))$ is limited to the 8-bit grayscale value.

$$I_{new}(x,y,z) = \min\{\max\{0, f(I(x,y,z)\}, 255\} \tag{2}$$

Thus, the histogram transformation is context-dependent and the initial wide range of values $[0, 6000]$ fits into the uint8=1byte data type (Figures 2 and 3).

2.3 CNN training

FCN is an encoder-decoder architecture (Figure 4), where the encoder obtains a convolutional code (small-sized feature) of the input image after several steps of halving and passing through the various usual layers (convolution, pooling, padding and batch normalization and dropout) in a CNN. The decoder is a symmetrical part of the encoder, built of transpose convolution, upsampling, concatenation and softmax layers. The encoder is the so-called contracting layer, since it reduces the image over a number of stages up to a given size. The output of the CNN at the end of the encoder is the code of the input

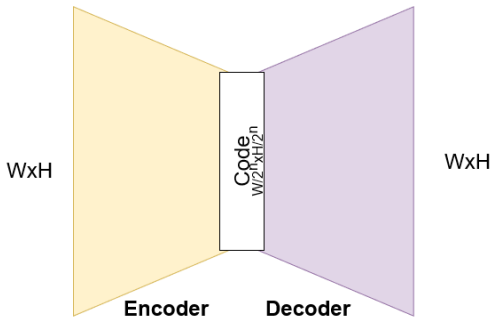


Figure 4: Encoder-decoder-type CNN

image. If the number of reductions is n, and each stage halves the input image size ($W \times H$), then after n steps the code size will be ($W/2^n \times H/2^n$). The decoder has to do the opposite operations in order to obtain a segmentation output of the same size as the original image. This is called the expending layer. The first FCN networks in the literature were SegNet [3], FCN [24] and U-Net [22].

The original U-Net [22] has 5 stages. From a grayscale input image of 572×572 pixels of resolution it obtains a code of 28×28 pixels. In the encoder part, each halving stage has the same structure: 2 convolution layers of kernel size 3×3 with a ReLU activation function (Figure 5 blue arrow), followed by a max-pooling layer with the role of halving the output of the previous convolution (Figure 5 dark red arrow). After each convolution, the input image size is reduced by 2 pixels in both width and height, due to the nature of the convolution operation applied. The inputs and outputs before and after each conv-layer are the stage-wise feature maps. At the first level of original size, the conv-layers are of depth 64 and 64; next, at a size of $1/2$, the convolutional layers are double to 128, 128, followed by two of depth 256, 256. At the size of $1/2^4$, the depth is doubled again to 512 and 512. At the end of the 5th stage, there are the last two convolutional layers of a depth of 1024, and the output of the last conv-layer is, in fact, the code of the input of size 28×28 pixels. From the code, the segmentation is obtained via the decoder part. Here, the convolutions are substituted by up-convolutions, also called transposed convolutions. These operations are the inverse-convolutions that are capable of doubling their input feature maps by padding them with zeros and passing through the kernel of the same size as the matching stage convolution kernel. In this manner they reach the $2 \times size$ (Figure 5 green arrow). The output of transposed convolutions have the same depth and size as the

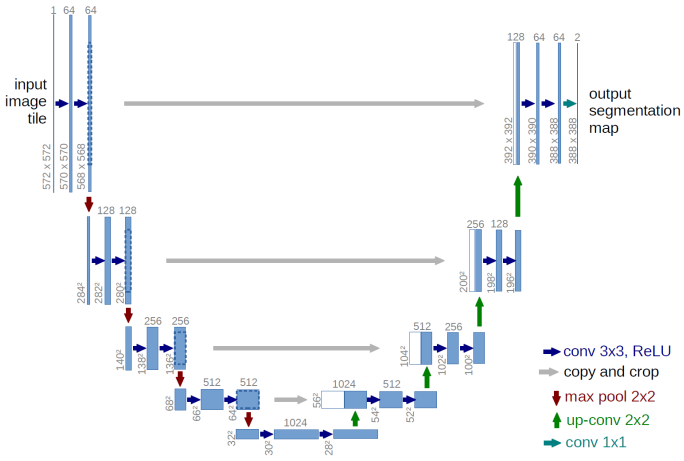


Figure 5: Original U-Net architecture [22]

corresponding convolutions. The trick here is the concatenation of the output of each encoder-stage to the first deconvolution at the same stage of the decoder (Figure 5 light gray arrow). Thus, if the input depth of the convolutions and deconvolutions were d_1 and d_1, respectively, the depth of concatenated layer will be $2d_1$. Consequently, the first deconvolution layer depth is doubled. The second deconvolution at each stage remains of the same depth d_1. The last layer is a bottleneck convolution (Figure 5 turquoise arrow) of depth 2 which obtains a segmentation map for both classes (object of interest and background.) All the details are shown in Figure 5.

In our adapted variant of U-Net, we consider 3 stages. The input image measures $240 \times 240 \times 4$ pixels. The third dimension is 4, containing the 4 image modalities (T1, T1c, T2 and FLAIR). Every stage consists of 2 convolutional layers. Each convolution is followed by batch-normalization and dropout layers. The stage-wise depth of the conv-layers is 32, 64 and 128. Halving is obtained via MaxPooling2D. The code size in our case is 60×60 pixels. The decoder part is the corresponding extending layer with the concatenation of the corresponding convolution output to the first deconvolution layer in each stage. The upsampling from a given size to its double, up to the original image size, is computed via transpose convolution with the role of Upsampling2D. The component diagram of the adapted U-Net architecture is shown in Figure 6.

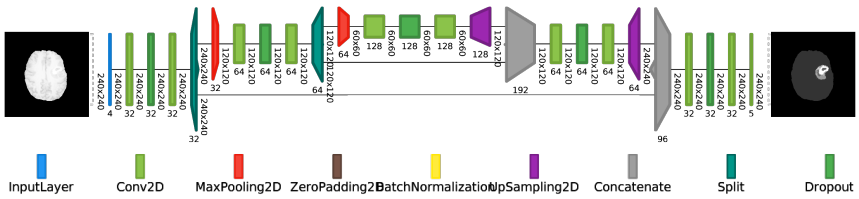


Figure 6: Adapted implementation of U-Net architecture for brain tumor segmentation

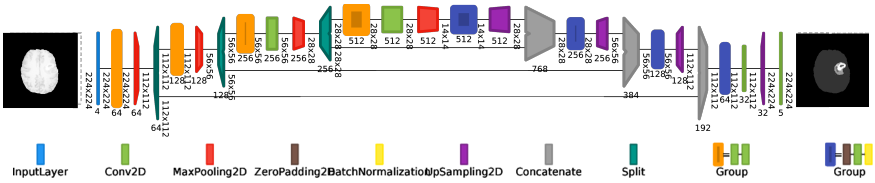


Figure 7: Adapted implementation of VGG-UNet architecture for brain tumor segmentation

The next architecture that was adapted into a U-Net-like architecture was the VGG16 [25] network. Here, the encoder part is formed based on the VGG16 architecture.

The VGG16-architecture [25] is made up of 5 stages from 224×224 pixels up to 7×7 pixels ($W/2^5 \times H/2^5$). Stages 1 and 2 contain 2 conv-layers of depth 64, 64 and 128, 128. Stages 3, 4 and 5 contain 3 conv-layers each of depth 256, 256, 256; 512, 512, 512 and 512, 512, 512. Here, the code size is 7×7 pixels. In case of classification, the code output is followed by two fully connected layers and a softmax classification layer.

The advantage of VGG16-UNet is that pretrained weights can be loaded onto this architecture. In the literature, the preloading of pretrained weights for other purposes is called transfer learning. We load the pretrained weights of ImageNet [30] into the encoder part and start the training process for the brain segmentation from those initial weight values. The ImageNet weights are weights obtained from training the VGG16 CNN network for distinguishing 1000 usual object classes, they are not related to segmentation.

The VGG16-UNet architecture, adapted for brain tumor segmentation, has the following structure: the adapted VGG16-UNet architecture. It considers the center-cropped brain images measuring 224×224 pixels. We consider 4 stages up to a size of 14×14 pixels. In one stage, we have two conv-layers padded to the same size as the stage input followed by the ReLU activation. In

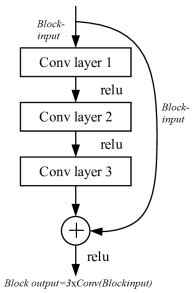


Figure 8: Residual Block with 3 conv-layers

Figure 7, a stage is denoted by the orange-colored Group. The corresponding transpose convolution is the blue group. The architectures of Figures 6 and 7 were created by the Net2Vis visualization grammar module [8].

The last architecture adapted for the purpose of multi-modal brain tumor segmentation was the ResNet50 [14]. The ResNet50 is made up of 5 convolutional blocks on the 5 consecutive sizes of the encoding part. Each block is implemented via the Residual module that learns the difference between the input and the output of the block. This residue is obtained by passing the input value through 2 or 3 convolutional layers. In our case the number of conv-layers of a residual block is 3 (Figure 8). The identity connection is in fact a skip connection.

Block1 is used once at the beginning, block2 is repeated 3 times with the 3 conv-layers $(1 \times 1, 64)$, $(3 \times 3, 64)$, $(1 \times 1, 256)$, block3 is repeated 4 times $(1 \times 1, 128)$, $(3 \times 3, 128)$, $(1 \times 1, 512)$, block4 appears 6 times $(1 \times 1, 256)$, $(3 \times 3, 256)$, $(1 \times 1, 1024)$ and the last block appears 3 times $(1 \times 1, 512)$, $(3 \times 3, 512)$, $(1 \times 1, 2048)$. The ResNet50-UNet is a network based on the ResNet architecture and combined with U-Net. The Encoder part of U-Net is substituted by the ResNet50 blocks. The architecture of ResNet50 is shown in Figure 9.

The convolution blocks represent the different stages on different input sizes. The triplets inside the conv-block represent the kernel-size $k \times k$ and the depth of the conv-layers $(k \times k, depth)$. Each conv-block is repeated a given number of times represented above the block, and the output size of every block is the resolution written under the block name.

The adapted ResNet50-UNet variant has corresponding transpose convolution layers concatenated to the outputs of the same-sized convolution block and followed by 2 convolution layer of depths (512,512); (256,256); (128,128)

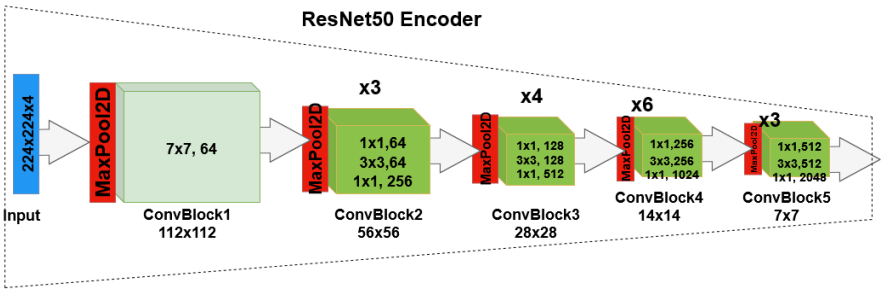


Figure 9: ResNet50 encoder: the first part of ResNet-UNet architecture

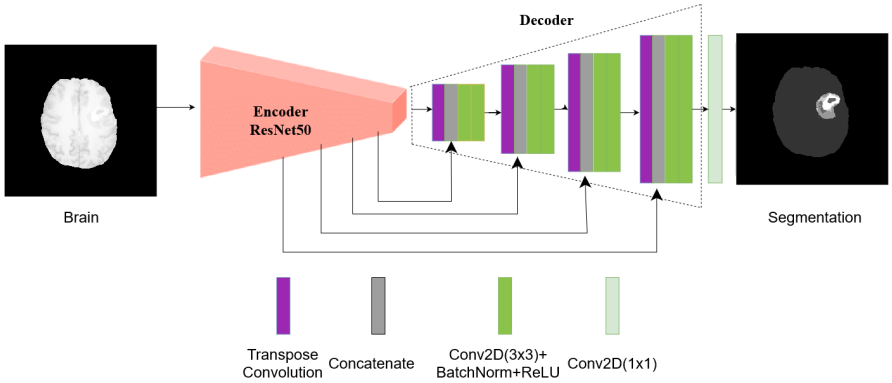


Figure 10: ResNet-UNet architecture

and (64,64). The output segmentation is obtained after a bottleneck convolution (Figure 10).

2.4 The training process

The training process is the usual training of CNN networks via the Backpropagation algorithm. The hyperparameters of the training are crucial from the perspective of the final training, validation and test scores.

Before the training starts, the preprocessed database is split into 3 parts: training (203 images=18676 slices)-60%, validation set (66 images=6006 slices)-20% and test set (66 images=10230 slices)–20%.

For the training and validation images, we only considered non-zero slices, while the test set used all slices. There were 18676 training image slices, 6006 validation images slices and 10230 test slices.

The main hyperparameters were set experimentally to obtain good validation performance. The batch size that fits into GPU memory is between 8 and 24 images. Such a batch of images is processed in parallel through the forward and backward passes of the CNN. The update to the weights is made once a batch. In every epoch, the whole training dataset is passed through the network once, in batches. The total number of training images over the batch size represents the number of iterations. Weights are adapted in every iteration. The entire process ran for 100 epochs or until a given validation Dice coefficient was met. The optimization method was AdaDelta and the loss function was the weighted mean Dice loss.

The AdaDelta optimization algorithm, instead of considering all past squared gradients, only considers those of a range with a fixed window size w. The sum of gradients is a recursive expression, an exponentially decaying average of all past squared gradients in w. It depends on the previous running average with weight γ and the current gradient with weight $(1-\gamma)$. The gradient update will be a product of the gradient and the proportion of the root-mean-square error (RMSE) of the previous update and the RMSE of the current gradient. The main advantage of AdaDelta is that the initial learning rate does not have to be set.

The loss based on which the optimization function was minimized is the weighted mean Dice-Sørensen loss (eq. 8). The Dice Similarity Coefficient (Dice) measures the similarity between two discrete pieces of data. In our case, it measures the similarity between the automatic segmentation obtained by our system and the gold standard annotations of experts. The Dice score is computed for every image in a batch, namely, for every image in the training set. It is a measure of similarity between the segmented image (Seg) and the ground truth (GT).

$$\mathrm{Dice}_I = \frac{2|\mathrm{Seg}_I \cap \mathrm{GT}_I|}{|\mathrm{Seg}_I| + |\mathrm{GT}_I| + \epsilon} \tag{3}$$

In the case of multi-class classification, the Dice score is computed class-wise, and the optimization process considers the mean-Dice loss.

$$\mathrm{mDice}_I = \sum_{c \in \mathrm{Classes}} \mathrm{Dice}_I^c \tag{4}$$

The mean Dice score on every image or on a batch is computed by

$$\mathrm{mDice} = \sum_{\mathrm{I} \in \mathrm{Images}} \mathrm{Dice}_{\mathrm{I}} \tag{5}$$

The loss is computed from the mean Dice score of a batch.

$$\mathrm{Dice}_{\mathrm{Loss}} = 1 - \mathrm{mDice}_{\mathrm{I}} \tag{6}$$

In our training we have tried to train the CNN network not only based on the simple mean-Dice, but on the class-wise weighted Dice score ($\mathtt{weight}_c$) as well.

$$w\mathrm{mDice}_{\mathrm{I}} = \frac{2 \sum_{c \in \mathrm{Classes}} \mathrm{weight}_c |\mathrm{Seg}_{\mathrm{I}}^c \cap \mathrm{GT}_{\mathrm{I}}^c|}{\sum_{c \in \mathrm{Classes}} \mathrm{weight}_c \left(|\mathrm{Seg}_{\mathrm{I}}^c| + |\mathrm{GT}_{\mathrm{I}}^c|\right)} \tag{7}$$

$$w\mathrm{mDice} = \sum_{\mathrm{I} \in \mathrm{Images}} \mathrm{m}w\mathrm{Dice}_{\mathrm{I}} \tag{8}$$

The role of implementing the weighted mean Dice score ($w\mathrm{mDice}$) is to put more emphasis on the classes with much fewer pixels in the dataset and to overcome the problem of class imbalance. The weights were computed to be inversely proportional to the average of number of pixels in each class. The total number of pixels in an image is $240 \times 240 \times 155 \approx 9$ million. Of these, 1.5 million are brain pixels. About 2,000-90,000 pixels is tumor tissue, of which 40-80% is edema, 15-60% is enhancing tumor and 0-5% represents necrosis.

2.5 Post-processing

In the final post-processing step, we considered the formal structure of the tumors, namely, the tumor is a compact nodule. Usually, there can be 1 or sometimes 2 large nodules. Thus, we considered the 2 largest nodules, being, in volume, more than 90% of the total detected tumor regions. This way, the small false positive pixel regions could be eliminated. Another step was the elimination of false detected background pixels inside the whole tumor. These are very dark gray parts that can be easily mistaken for the background class by every autonomous system, but they are in fact necrotic regions.

3 Results and discussion

In our research we have implemented 3 variants of the U-Net [22] convolutional neural network and adapted them to be suitable for brain tumor seg-

mentation. The training of the autonomous system is based on the supervised learning method. The initial BraTS dataset is split into 3 disjoint sets: training, validation and test sets. The convolutional networks are trained on 18676 multi-modal images. The network is optimized via the AdaDelta optimization method, considering the weighted Dice loss function as optimization criteria. The weights are crucial in order to overcome the drawbacks of a class-unbalanced dataset. The weight multiplication factor for class 0 is $m_{w_0} = 1$, the weight for class 1 is $m_{w_1} = 2000$, while classes 2 and 4 are assigned $m_{w_2} = 400$ and $m_{w_4} = 800$, respectively.The factors are inversely proportional to the average size of the tumor tissue of each class. These weights are converted into the [0,1] interval.

$$weight_c = \frac{m_{w_c}}{\sum_{c \in Classes} m_{w_c}} \tag{9}$$

These weights are used only in computation of training losses. The training process stops after 100 epochs or if the mean Dice score on validation reaches 0.9915. Validation mean Dice score was determined experimentally. This value assures the sufficiently high Dice score (above 80-85%) on the whole tumor. A mean Dice score of 0.992-0.993 can be reached after running additional tens or hundreds of epochs. The further training last very long compared to the benefit of $< 1\%$ on the overall validation mean Dice score. The hyperparameters for training are batch-size between 16-24 images, and the number of iterations for an epoch is 500-1200. The weight parameter in AdaDelta is $\gamma = 0.9$. The total training time of a single CNN architecture is between 36 and 72 hours.

The hardware used for training was an NVIDIA 1080Ti GPU card with 11GB of memory. The training was done in Tensorflow Keras in the Python programming language. The visualization of the histograms and pre- and post-processing steps were done in ImageJ and Matlab.

The total number of parameters, namely, the trained and untrained weights are shown in Table 1.

Figures 11 and 12 show the comparison of the training process for the 3 convolutional neural networks.

The training process is very similar from the perspective of all 3 network types. It can be seen in Figure 11 that the simple U-Net increases the most, only after a few epochs. The mean Dice score indicates similarity as well, but the training loss for U-Net and ResNet50-UNet decreases more than for VGG-UNet.

Table 2 shows the increase in mean Dice scores throughout the training process. U-Net and ResNet50-UNet are the two best from this perspective.

CNN	Total parameters	Non-trainable parameters	Trainable parameters
U-Net	4,473,477	3,328	4,470,149
VGGUNet	13,324,667	1,920	12,322,757
ResNet50-UNet	16,375,173	32,512	16,342,661

Table 1: Trainable parameters of CNNs

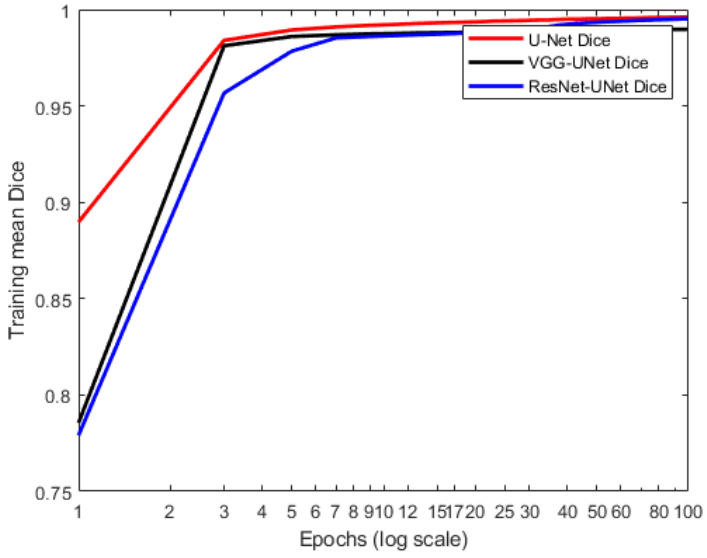


Figure 11: Training mean Dice score

Epoch	**U-Net**		**VGG-UNet**		**ResNet-UNet**	
	Loss	**Dice**	**Loss**	**Dice**	**Loss**	**Dice**
5	0.1102	0.8898	0.2144	0.7856	0.2209	0.7791
10	0.0082	0.9918	0.0126	0.9874	0.0140	0.9860
20	0.0065	0.9935	0.0118	0.9882	0.0121	0.9879
30	0.0057	0.9943	0.0114	0.9886	0.0102	0.9898
40	0.0051	0.9949	0.0111	0.9889	0.0078	0.9922
50	0.0048	0.9952	0.0109	0.9891	0.0067	0.9933
60	0.0046	0.9954	0.0108	0.9892	0.0062	0.9938
70	0.0043	0.9957	0.0107	0.9893	0.0056	0.9944
80	0.0041	0.9959	0.0106	0.9894	0.0053	0.9947
90	0.0040	0.9960	0.0103	0.9897	0.0051	0.9949
100	0.0039	0.9961	0.0102	0.9898	0.0049	0.9951

Table 2: Loss and Dice scores in training

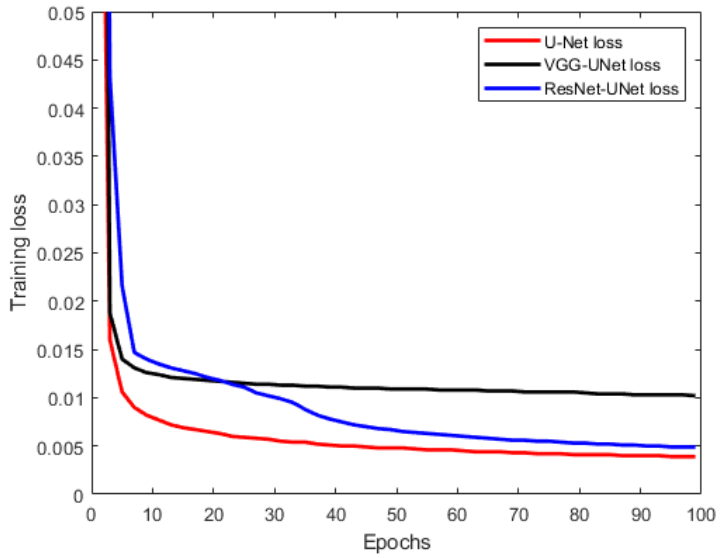


Figure 12: Training loss

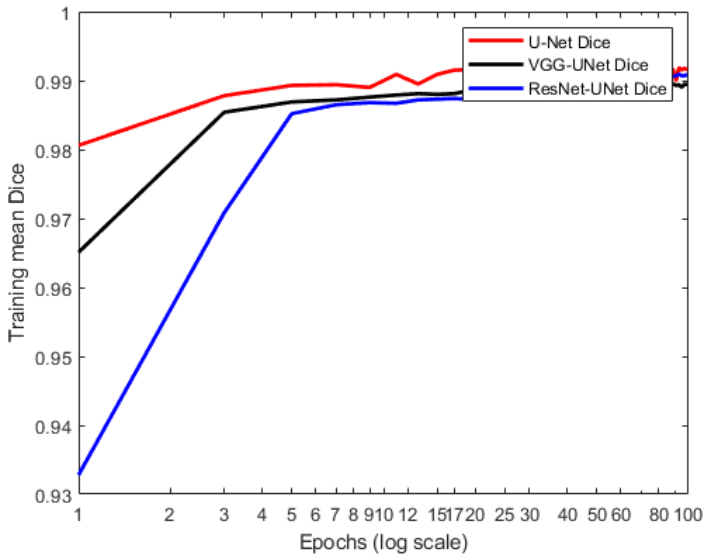


Figure 13: Validation mean Dice score

Form the standpoint of an autonomous systems the generalization capability is essential. A CNN is capable of generalization i.e. recognizes non-seen images with the same precision as in training if the validation score on the unseen validation set is similar to those obtained on the training set. Figures 13 and 14 and Table 3 present the results on the validation set.

The mean Dice score is very similar for all 3 networks. A very small change in the mean Dice score has a great impact on class-wise Dice scores. If the mean Dice score is not over 0.9915, then the segmentation of different classes is not high enough.

The real segmentation accuracy of the trained CNNs is determined by the performance scores of the different classes.

Tables 4 and 5 show the class-wise performance measure of the Dice score on the training and test sets. The optimization of the CNN models were done not on the class-wise Dice scores, but on the weighted mean Dice score. Small changes in the weighted mean Dice score may lead to noticeable class-wise Dice score change.

These tables show a superior performance for the simple U-Net and ResNet-UNet. The background and healthy tissues are detected with a Dice score of 0.9984-0.9985. Class1+3 represent necrosis and the non-enhanced tumor,

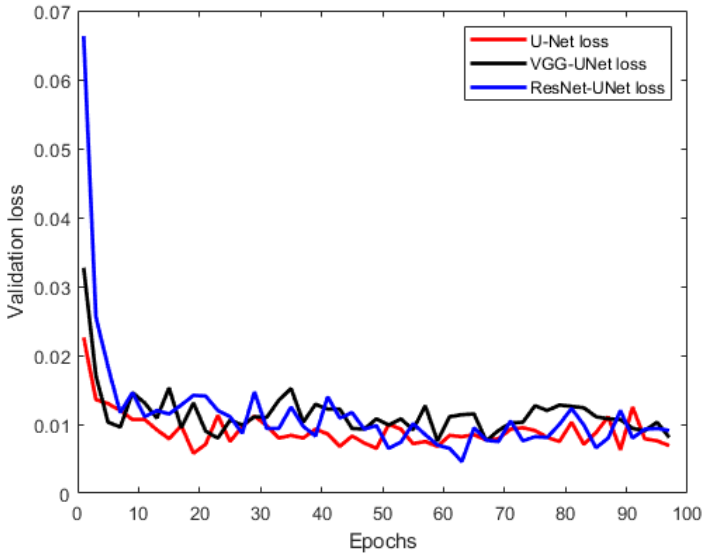


Figure 14: Validation loss

Epoch	**U-Net**		**VGG-UNet**		**ResNet-UNet**	
	Loss	**Dice**	**Loss**	**Dice**	**Loss**	**Dice**
5	0.0226	0.9806	0.0327	0.9651	0.0663	0.9328
10	0.0107	0.9890	0.0146	0.9876	0.0146	0.9868
20	0.0058	0.9916	0.0131	0.9885	0.0142	0.9873
30	0.0112	0.9904	0.0111	0.9885	0.0147	0.9885
40	0.0093	0.9914	0.0129	0.989	0.0083	0.9906
50	0.0065	0.9918	0.0108	0.9887	0.0098	0.9905
60	0.0075	0.9917	0.0127	0.9888	0.0085	0.9905
70	0.0079	0.9918	0.0091	0.9895	0.0075	0.9908
80	0.0075	0.9916	0.0128	0.989	0.0100	0.9907
90	0.0063	0.9916	0.0107	0.9895	0.012	0.9906
100	0.0074	0.9915	0.0102	0.9894	0.0108	0.9909

Table 3: Loss and Dice scores for the validation set

Training	**Dice-Cl0**	**Dice-Cl1+3**	**Dice-Cl2**	**Dice-Cl4**	**Dice-WT**	**Dice-TC**
U-Net	0.9991	0.7153	0.8557	0.8113	0.9200	0.8793
VGG-UNet	0.9981	0.4785	0.7237	0.7088	0.8219	0.7704
ResNet-UNet	0.9991	0.6989	0.8495	0.7860	0.9186	0.8722

Table 4: Class-wise Dice scores in training

Test	**Dice-Cl0**	**Dice-Cl1+3**	**Dice-Cl2**	**Dice-Cl4**	**Dice-WT**	**Dice-TC**
U-Net	0.9984	0.474	0.7272	0.7338	**0.8359**	**0.7773**
VGG-UNet	0.9984	0.4156	0.7117	0.7334	**0.8217**	**0.7618**
ResNet-UNet	0.9985	0.5005	0.7372	0.7488	**0.8434**	**0.7865**

Table 5: Class-wise Dice scores for the test set

respectively. This class can be easily mistaken for the dark background because of its very dark intensity on the MRI image. The Dice score results for this class are the lowest (47%-50%). Compared to the difficult class, the other two classes Class2 and Class4 are detected with a higher Dice score of (73%-74%).

In the literature, it is not the class-wise Dice coefficients that are compared; nevertheless, from the perspective of a physician, the Whole Tumor (WT), the Tumor Core (TC) and the Enhanced Tumor (ET) are much more important. The WT is the joint region of all the tumor types (Class1+3 $\cup$ Class2 $\cup$ Class4). The TC is the union of Class1+3 $\cup$ Class4, without the edema region. The third type ET is Class4.

If we consider these tumoral regions, the results become much better. Tables 6 and 7 show the results for WT and TC, and the results for ET are the same as those specified for Cl4 (Tables 4 and 5).

All the three U-Net models were improved via post-processing steps. The two main corrections steps applied were fillholes and connected components. In the 3D image if a region inside a tumor was detected background it was set to be Class1, i.e. necrosis. From the detection results only the greatest connected components were kept for which the cumulative sum of the total pixels is above 90% of the total detected tumor pixels. The rest (small regions compared to the volume of greatest) was considered false positive and was

Training	**Dice-WT**	**Sens-WT**	**Spec-WT**	**Dice-TC**	**Sens-TC**	**Spec-TC**
U-Net	0.9200	0.9278	0.999	0.8793	0.8820	0.9987
VGG-UNet	0.8219	0.9081	0.9972	0.7704	0.7395	0.9983
ResNet-UNet	0.9186	0.9432	0.9988	0.8722	0.8563	0.9989

Table 6: Dice score of tumor regions in training

Test	**Dice-WT**	**Sens-WT**	**Spec-WT**	**Dice-TC**	**Sens-TC**	**Spec-TC**
U-Net	0.8359	0.8335	0.9986	0.7773	0.8259	0.9977
VGG-UNet	0.8217	0.8524	0.9982	0.7618	0.7843	0.9980
ResNet-UNet	0.8434	0.8586	0.9984	0.7865	0.8122	0.9981

Table 7: Dice score of tumor regions for the test set

erased from the segmentation. These post-processing steps brought a 2-3% improvement compared to the original results (Tables 5 and 9).

The best results here are obtained by ResNet-UNet WT=0.8688 and TC=0.8286. We also considered the ensemble model, which is the joint response of the probability maps for the 3 individual networks and the ensemble. The joint response obtains a 1-2% improvement compared to the best out of the three models presented (Table 8).

Figure 15 shows the segmentation results for the three U-Net variants studied. Visual comparison of the results becomes exceedingly difficult. The results are very similar; however, U-Net and ResNet-UNet achieve the best results, especially on Class1+3. On Figure 15 different colors denote different tissues: background (Class0) is black, Class2 is the edema denoted by light gray, Class4

Test	**Dice-WT**	**Sens-WT**	**Spec-WT**	**Dice-TC**	**Sens-TC**	**Spec-TC**
Ensemble model	0.8765	0.8727	0.9988	0.8169	0.8461	0.9981

Table 8: Dice score of ensemble model

Test	Dice-WT	Sens-WT	Spec-WT	Dice-TC	Sens-TC	Spec-TC
U-Net	0.8623	0.8542	0.9986	0.7992	0.8496	0.9976
VGG-UNet	0.8534	0.8660	0.9983	0.7892	0.8276	0.9978
ResNet-UNet	0.8688	0.8836	0.9984	0.8076	0.8286	0.9981

Table 9: Dice score of tumor regions with post-processing

BraTS 2020	Dice-WT	Dice-ET	Dice-TC
McHugh [19]	0.859	0.766	0.788
Savadikar [23]	0.818	0.688	0.716
Ali [1]	0.871	0.748	0.748
Bommineni [7]	0.883	0.718	0.787
Isensee [15]	0.889	0.820	0.850
Our ResNet-UNet	0.8688	0.7507	0.8076
Our Ensemble	0.8765	0.7588	0.8169

Table 10: Comparison to BraTS2020

enhanced tumor is the white part, and the innermost tumor region is the non-enhanced+necrotic part (Class1+3) labelled with light gray.

Table 10 shows the most up-to-date results published for the BraTS2020 competition in 2021. The models featured all are different version of the U-Net network. Our WT, ET and TC results are similar to the results described in the literature.

4 Conclusions

In this paper we present an automated system used for brain tumor segmentation. We compare three well-known CNN networks used for object detection and adapt them for the purpose of brain tumor segmentation. The three net-



(a) Ground truth (b) U-Net (c) VGG-UNet (d) ResNet-UNet (e) Ensemble

Figure 15: Segmented images – black: background; light gray: edema; white: enhanced tumor; dark gray: non-enhanced tumor+necrosis

works converted to encoder-decoder networks were U-Net, VGG16-UNet and ResNet50-UNet. The main contributions of this paper are: the intensity uniformization of the BraTS2020 dataset, the adaptation of CNN networks for brain tumor segmentation, training and evaluation of the segmentation system obtained. In the preprocessing phase all of the different image intensity ranges were transformed into the standard grayscale domain of [0,255] while maintaining the histogram form of the original image. Using this preprocessed dataset we have adapted and trained the mentioned three models for brain tumor segmentation. Moreover, these models were combined into an ensemble that achieved an overall performance boost of 2%.

To obtain even better results, we further suggest to implement and apply other types of CNN networks that use the attention module and multi-pathway models with different scales and resolutions with the role of correcting the segmentation on the tissue borders. We also propound to implement the 3D U-Net for brain tumor segmentation and compare the results presented to those of a 3D-CNN. The 3D models require more GPU capacity to fit into the memory and are usually implemented in HPC environments. In addition, a cascade of multiply applied binary classifications of the different tumor regions would improve segmentation. In our future work, we suggest to first segment the whole tumor, followed by the edema, tumor core and finally the enhanced tumor and necrosis. The role of this type of classification is an even better delimitation of different tissue types. Moreover, we set to improve our results even more by concentrating especially on the tissue borders that are the least accurately detected. All these suggestions would additionally improve the results presented in this paper.

Acknowledgements

The work was supported by the University of Medicine, Pharmacy, Science and Technology "George Emil Palade" of Târgu Mureș, Research Grant Number 292/4/14.01.2020.

References

[1] M. J. Ali, M. T. Akram, H. Saleem, B. Raza, A. R. Shahid, Glioma segmentation using ensemble of 2D/3D U-nets and survival prediction using multiple features fusion, *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, *Lecture Notes in Comput. Sci.*, **12659** (2021), 189–199. ⇒70

[2] U. Baid, S. Ghodasara, M. Bilello et al. The RSNA-ASNR-MICCAI BraTS 2021 benchmark on brain tumor segmentation and radiogenomic classification, *CoRR*, abs/2107.02314, 2021. ⇒51

[3] V. Badrinarayanan, A. Kendall, R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, abs/1511.00561, 2015. ⇒56

[4] S. Bakas, M. Reyes, A. Jakab et al. Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge, *CoRR*, abs/1811.02629, 2018. ⇒51

[5] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, J. Freymann, K. Farahani, C. Davatzikos, Segmentation labels for the pre-operative scans of the TCGA-GBM collection, 2017. ⇒52

[6] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, J. Freymann, K. Farahani, C. Davatzikos, Segmentation labels for the pre-operative scans of the TCGA-LGG collection, 2017. ⇒52

[7] V. L. Bommineni, Piecenet: A redundant Unet ensemble,*Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, *Lecture Notes in Comput. Sci*, **12659** (2021), pp. 331–341. ⇒70

[8] A. Bäuerle, C. van Onzenoodt, T. Ropinski, Net2vis – a visual grammar for automatically generating publication-tailored CNN architecture visualizations, *IEEE Transactions on Visualization and Computer Graphics*, **27**, 6,(2021) 2980–2991. ⇒59

[9] S. Chen, C. Ding, M. Liu, Dual-force convolutional neural networks for accurate brain tumor segmentation, *Pattern Recognition*, **88** (2019), 90–100. ⇒51

[10] C.-B. S. Sang-Geun Choi, Detection of HGG and LGG brain tumors using U-Net,*Medico Legal Update*, **19**, 1, (2019) 560–565. ⇒50

[11] Y. Ding, C. Li, Q. Yang, Z. Qin, Z. Qin, How to improve the deep residual network to segment multi-modal brain tumor images, *IEEE Access*, **7**, (2019) 152 821–152 831. ⇒51

[12] N. Gordillo, E. Montseny, P. Sobrevilla, State of the art survey on MRI brain tumor segmentation, *Magnetic Resonance Imaging Journal*, **31**, 8, (2013) 1426–1438. ⇒51

[13] A. Győrfi, L. Szilágyi, L. Kovács, A fully automatic procedure for brain tumor segmentation from multi-spectral MRI records using ensemble learning and atlas-based data enhancement, *Applied Sciences*, **11**, 2 (2021). ⇒54

[14] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, *CoRR* abs/1512.03385, 2015. ⇒59

[15] F. Isensee, P. F. Jäger, P. M. Full, P. Vollmuth, K. H. Maier-Hein, nnU-net for brain tumor segmentation, *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries.Lecture Notes in Comput. Sci.*, **12659** (2021), 118–132. ⇒51, 70

[16] K. Kamnitsas, C. Ledig, V. F. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, B. Glocker, Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation, *Medical Image Analysis*, **36**, (2017) 61–78. ⇒51

[17] Kőble, A., Győrfi, Á., Csaholczi, S., Surányi, B., Dénes-Fazakas, L., Kovács, L., Szilágyi, L. Identifying the most suitable histogram normalization technique for machine learning based segmentation of multispectral brain MRI data. *2021 IEEE AFRICON, Arusha, Tanzania*, 2021, pp. 71-76. ⇒54

[18] L. Lefkovits, S. Lefkovits, L. Szilágyi, Brain tumor segmentation with optimized random forest, *International Workshop on Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries.Lecture Notes in Comput. Sci.* **10154** (2016), 88–99. ⇒51

[19] H. McHugh, G. M. Talou, A. Wang, 2D dense-unet: A clinically valid approach to automated glioma segmentation, *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, *Lecture Notes in Comput. Sci.* **12659** (2021), 69–80. ⇒70

[20] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer et al. The multimodal brain tumor image segmentation benchmark (brats), *IEEE Transactions on Medical Imaging*, **34**, 10 (2015) 1993–2024. ⇒51

[21] S. Pereira, A. Pinto, V. Alves, C. A. Silva, Brain tumor segmentation using convolutional neural networks in MRI images, *IEEE Transactions on Medical Imaging*, **35**, 5 (2016) 1240–1251. ⇒51

[22] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation,*CoRR*, abs/1505.04597, 2015. ⇒56, 57, 62

[23] C. Savadikar, R. Kulhalli, B. Garware, Brain tumour segmentation using probabilistic U-Net,*Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, *Lecture Notes in Comput. Sci.* **12659** (2021), pp. 255–264. ⇒70

[24] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**, 4, (2017) 640–651. ⇒56

[25] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, *CoRR* abs/1409.1556, 2015. ⇒58

[26] D. Wu, Y. Ding, M. Zhang, Q. Yang, Z. Qin, Multi-features refinement and aggregation for medical brain segmentation, *IEEE Access*, **8** (2020), 57 483–57 496. ⇒51

[27] Y. Xue, T. Xu, H. Zhang, L. R. Long, X. Huang, SegAN: Adversarial network with multi-scale L1 loss for medical image segmentation, *Neuroinformatics*, **16**, 3-4, (2018), 383–392. ⇒51

[28] X. Zhao, Y. Wu, G. Song, Z. Li, Y. Zhang, Y. Fan, A deep learning model integrating FCNNs and CRFs for brain tumor segmentation, *Medical Image Analysis*, **43**, (2018) 98–111. ⇒51

[29] ∗∗∗ Brain Tumor Segmentation (BraTS) challenge, https://www.med.upenn.edu/cbica/brats2021/, 2021, online; accessed, April 2022. ⇒51

[30] ∗∗∗ ImageNet,https://image-net.org, 2022, online; accessed, April 2022. ⇒58

[31] ∗∗∗ Medical Segmentation Decathlon Challenge, http://medicaldecathlon.com/, 2021, online; accessed, April 2022. ⇒52

Received: April 20, 2022 • Revised: May 9, 2022