Variational-Autoencoder Regularized 3D MultiResUNet for the BraTS 2020 Brain Tumor Segmentation

Jiarui Tang[1], Tengfei Li[1], Hai Shu[2], and Hongtu Zhu[1](✉)

[1] University of North Carolina, Chapel Hill, NC, USA
htzhu@email.unc.edu
[2] Department of Biostatistics, School of Global Public Health, New York University, New York, NY 10003, USA

Abstract. Tumor segmentation is an important research topic in medical image segmentation. With the fast development of deep learning in computer vision, automated segmentation of brain tumors using deep neural networks becomes increasingly popular. U-Net is the most widely-used network in the applications of automated image segmentation. Many well-performed models are built based on U-Net. In this paper, we devise a model that combines the variational-autoencoder regularuzed 3D U-Net model [10] and the MultiResUNet model [7]. The model is trained on the 2020 Multimodal Brain Tumor Segmentation Challenge (BraTS) dataset and predicts on the validation set. Our result shows that the modified 3D MultiResUNet performs better than the previous 3D U-Net.

Keywords: Brain tumor segmentation · 3D U-Net · MultiResUNet · Variational-autoencoder regularization

1 Introduction

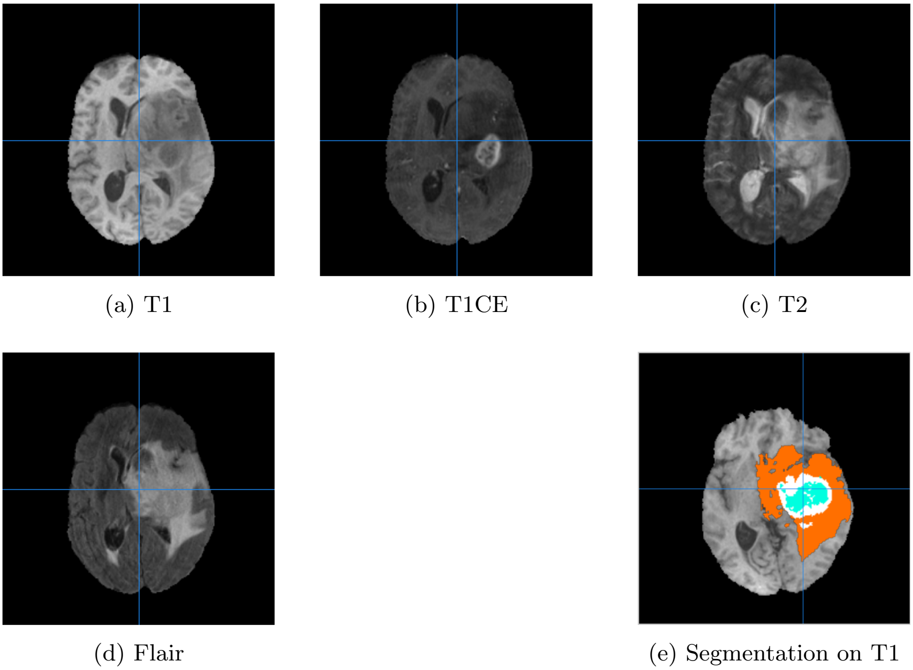
Multimodal MRI scans are useful to identify brain tumors, and the brain tumor segmentation is an important task before the diagnosis. The major goal of the medical image segmentation is to extract the areas of interest in an image, such as tumor regions. It allows doctors to focus on the most important areas for diagnosis or monitoring [7,11]. For a long time, manual tumor segmentation from MRI scans conducted by physicians is a time-consuming task. In order to speed up the process of image segmentation, many automated methods were developed [2–5,7,9,10]. With the rapid development of deep learning technologies in the field of computer vision, deep-learning based automated brain tumor segmentation becomes increasingly popular [14]. In particular, convolutional neural networks (CNNs) show great successes in image segmentation tasks [12], and especially the U-Net [13] earns the most credits.

Multimodal Brain Tumor Segmentation Challenge (BraTS) is an annual challenge aims at gathering state-of-the-art methods for the segmentation of

A. Crimi and S. Bakas (Eds.): BrainLes 2020, LNCS 12659, pp. 431–440, 2021.
https://doi.org/10.1007/978-3-030-72087-2_38

brain tumors. Participants are provided with clinically acquired training data to develop their own models and produce segmentation labels of three glioma sub-regions: enhancing tumor (ET), tumor core (TC), and whole tumor (WT) [2–5,9,10]. The BraTS 2020 training dataset contains 369 cases, including 293 high-grade gliomas (HGG) and 76 low-grade gliomas (LGG) cases, each with four 3D MRI modalities: the native (T1) and the post-contrast T1-weighted (T1CE) images, and the T2-weighted (T2) and the T2 Fluid Attenuated Inversion Recovery (FLAIR) images. The example images of the four modalities are shown in Fig. 1. The validation dataset contains 125 cases and the test dataset contains 166 cases. The dimension of each MRI image is 240 × 240 × 155.

In this paper, we propose a model that combines the variational-autoencoder (VAE) regularized 3D U-Net model [10] and the MultiResUNet model [7], which is used to train end-to-end on the BraTS 2020 training dataset. Our model follows the encoder-decoder structure of the 3D U-Net model of [10] used in BraTS 2018 Segmentation Challenge but exchanges the ResNet-like block in the structure with the “MultiRes block” and connects the feature maps from the encoder


(a) T1

(b) T1CE

(c) T2

(d) Flair

(e) Segmentation on T1

Fig. 1. Visualization of the four modalities in BraTS 2020 training Dataset (a) T1, (b) T1CE, (c) T2 and (d) Flair, and (e) the segmentation on T1. These images are from the same patient. In (e), the blue area represents the Necrotic and Non-Enhancing Tumor (NCR/NET), the orange area represents the peritumoral edema (ED) and the white area represents the GD-enhancing tumor (ET). (Color figure online)

stages to the decoder stages with the “Res path”, a chain of convolutional layers with residual connections [7].

2 Related Work

2.1 VAE-regularized 3D U-Net

In BraTS 2018, the top-1 winner Myronenko [10] proposed an encoder-decoder CNN model with an asymmetrically large encoder to extract deep image features and a VAE branch to regularize the encoder. In BraTS 2019, the champion team Jiang et al. [8] proposed a novel model, a two-stage cascaded U-Net. The first stage of the model is a variant of the asymmetrical U-Net in [10] and the second stage of the model doubles the number of filters in the initial 3D convolution to increase the network width and has two decoders, one using deconvolutions and another using the trilinear interpolation to facilitate regularizing encoders. The top ranks of the two models in the challenge have proved the power of VAE-regularized 3D U-Nets in the MRI brain tumor segmentation.

2.2 MultiResUNet

In early 2020, Ibtehaz and Rahman [7] proposed a modified U-Net, called MultiResUNet, which outperforms the classical U-Net. There are two important parts of the architecture: the MultiRes block and the Res path. In the MultiRes block, there are three 3 × 3 consecutive convolutional layers with gradually increasing numbers of filters. A residual connection and a 1 × 1 convolutional layer are added in each MultiRes block to gather more spatial information. The Res path passes the feature maps from the encoder stage through a chain of 3 × 3 filters with 1 × 1 filters residual connections and then concatenates them with the decoder features. This MultiResUNet can also be applied to 3D images. The structures of the MultiRes block and the Res path are shown in Figs. 2 and 3, respectively.

In this paper, we modify the VAE-regularized 3D U-Net model [10] by exchanging the initial 3 × 3 × 3 3D convolution block with a revised 3D MultiRes block based on the MultiRes block in [7] and adding the Res path between the encoder and decoder stages.

3 Method

The architecture of our proposed network is shown in Fig. 4. Due to the limited computational resource, each image is cropped from the size of 240 × 240 × 155 voxels to 160 × 192 × 128 and then resized to 80 × 96 × 64. Each MRI modality is fed into the network as one channel and so we have totally four channels.

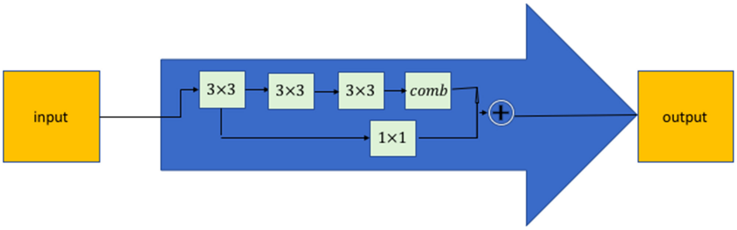


Fig. 2. The structure of the MultiRes block: the green boxes represent convolutional layers. The three 3 × 3 convolutional layers have increasing numbers of filters and then they are concatenated together, which can help the model capture different information from different scales. Then the result from the 1 × 1 convolutional layer following the input is added to the result from the three 3 × 3 convolutional layers to obtain the final output of the MultiRes block. (Color figure online)

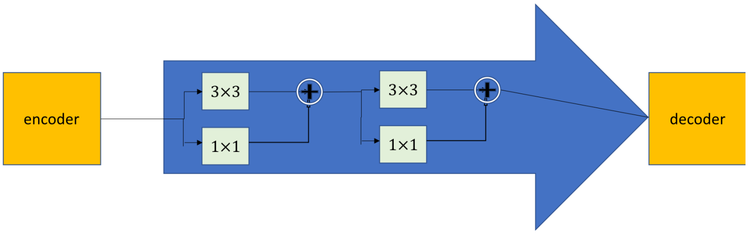


Fig. 3. The structure of the Res path: the green boxes represent convolutional layers. In many classic U-Nets, the encoder feature maps are simply sent to the decoder part. However, it is possible that there is semantic gap between the features of the decoder part and the encoder part, since the decoder part experiences more processing compared to the encoder part. Therefore, Ibtehaz and Rahman [7] proposed the Res path between the encoder and decoder before the features from the encoder part are concatenated to the decoder part. Within the Res path are series of 3 × 3 convolution layers with 1 × 1 residual connections. (Color figure online)

3.1 Encoder

The encoder part has multiple MultiRes blocks, each of which contains three 3 × 3 × 3 sequential convolution layers with a 1 × 1 × 1 residual connection to gather spatial features at different scales. After each MultiRes block, we use strided convolutions to downsize the image dimension by 2 and increase feature size by 2. The number of filters in the three 3 × 3 × 3 sequential convolution layers is 6, 12 and 18 respectively. Each MultiRes block contains the Group Normalization and the ReLU activation function. The output layer uses the sigmoid activation function.

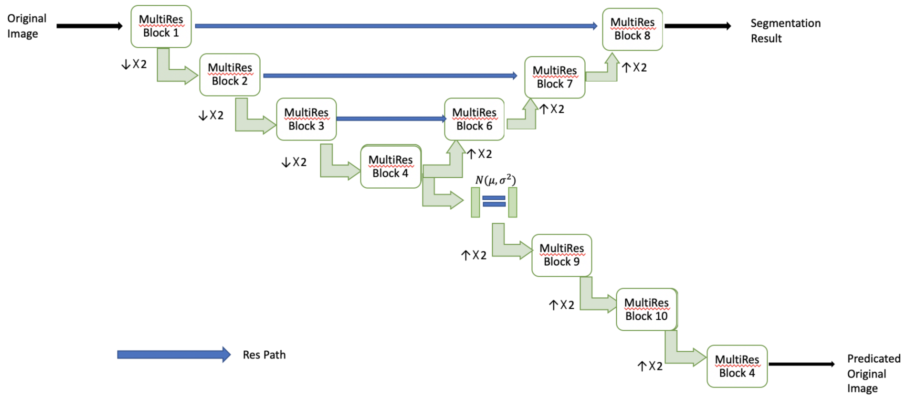


Fig. 4. The architecture of our MultiResUNet with a VAE branch. The size of the original input is 4 × 80 × 96 × 64. Each MultiRes block has three 3 × 3 × 3 sequential convolution layers and a 1 × 1 × 1 convolutional residual connection. Before each convolution layer, we use the group normalization and the ReLU activation function. In the Res path, there are two 3 × 3 × 3 convolutional layers each with one 1 × 1 × 1 residual connection. The output segmentation result has three channels and the shape of each channel is 4 × 80 × 96 × 64. Finally, we use the sigmoid activation function to obtain the segmentation for the three regions: WT (TC+ET), TC (ET+NCR/NET) and ET.

3.2 Decoder with a VAE Branch

The decoder part also has multiple MultiRes blocks. After the decoder, the output has three channels and the image of each channel has the same size as the input. We use strided convolutions to upsize the image dimension by 2 and decrease the feature size by 2.

For the VAE branch, we adopt the similar VAE structure used in [10]. The output of the fourth MultiRes block (MultiRes Block 4 in Fig. 4) is reduced to 256 × 1. We then draw a sample from a Gaussian distribution whose mean and standard deviation are both 128. Then, without connection to the encoder part through Res Path, we use the same structure as the decoder part to return to the original dimension 4 × 80 × 96 × 64. This step helps us to regularize encoder in training to prevent overfitting.

The VAE detail is shown in Table 1.

3.3 Res Path

The original skip connection between encoder and decoder parts is replaced by the Res path in our network model. Within each Res path, there is a chain of two 3 × 3 × 3 convolutional layers our network architecture and each convolutional layer has one 1 × 1 × 1 convoulational residual connection. The feature maps from the encoder parts is passed into the Res path and the output is concatenated

Table 1. The VAE part. Conv3 represents 3 × 3 × 3 convolutional layer. GN is the group normalization and the group size is 8. Conv1 represents 1 × 1 × 1 convolutional layer. Conv1 RC is the 1 × 1 × 1 convolutional residual connection in the MultiRes block. AddId is the addition of the residual connection. Concate is the concatenation of the three consecutive Conv3. Step 5, 7 and 9 are the MultiRes Block 9, 10, 11 in Fig. 4.

Step	Details	Output size
1	GN, ReLU, Conv(16), Dense(256)	256 × 1
2	Sample $N(128, 128^2)$	128 × 1
3	Dense, ReLU, Conv1, UpLinear	256 × 10 × 12 × 8
4	Conv1, UpLinear	128 × 20 × 24 × 16
5	GN, ReLu, Conv3, Conv3, Conv3, Concate, Conv1 RC, AddId	128 × 20 × 24 × 16
6	Conv1, UpLinear	64 × 40 × 48 × 32
7	GN, ReLu, Conv3, Conv3, Conv3, Concate, Conv1 RC, AddId	64 × 40 × 48 × 32
8	Conv1, UpLinear	32 × 80 × 96 × 64
9	GN, ReLu, Conv3, Conv3, Conv3, Concate, Conv1 RC, AddId	32 × 80 × 96 × 64
10	Conv1	4 × 80 × 96 × 64

with the decoder features. The number of filters in the Res Path reduces as the image downsizes. All convolutions in the Res path have 32 filters.

3.4 Loss Function

We use a combined loss function with three components: the soft Dice loss (L_{dice}), the L2 loss on the VAE part (L_{L2}) and the VAE penalty term (L_{KL}), KL divergence between the estimated normal distribution $N(\mu, \sigma^2)$ and the prior distribution $N(0, 1)$, which was used in [10].

The equation of the soft dice loss is

$$L_{dice} = \frac{2 * \sum S * R}{\sum S^2 + \sum R^2 + \epsilon} \tag{1}$$

where S represents the true labels and R represents the predicted output by the model.

The equation of L_{L2} is

$$L_{L2} = ||I_{origin} - I_{VAE_{pred}}|| \tag{2}$$

where I_{origin} is the original input image and $I_{VAE_{pred}}$ is the predicted image from the VAE part.

The equation of L_{KL} is

$$L_{KL} = \frac{1}{N} \sum \mu^2 + \sigma^2 - log\sigma^2 - 1 \tag{3}$$

where N is the number of voxels in the image, μ and σ are the parameters of the estimated normal distribution.

3.5 Optimization

We use the Adam optimizer with an initial learning rate of $\alpha_0 = 10^{-4}$ and decrease the learning rate by

$$\alpha = \alpha_0 * (1 - \frac{e}{N_e})^{0.9}, \tag{4}$$

where e is an epoch counter, and N_e is a total number of epochs. The total epoch is set to 300 and the batch size is 1.

4 Experiment

We use Python [15] to implement the experiment. Particularly, the library Keras [6] with Tensorflow [1] as the backend is used to build the network model. The model is trained on the BraTS 2020 training dataset (total 369 cases) without additional in-house data.

For the training dataset, there are 369 cases, among which 293 cases are high-grade gliomas (HGG) and 76 are low-grade gliomas (LGG) cases. There are 125 cases in the validation dataset. The validation result is submitted to CBICA's Image Processing Portal (https://ipp.cbica.upenn.edu) for evaluation.

4.1 Data Preprocessing and Augmentation

All the original images are preprocessed to have mean 0 and standard deviation 1. The image is randomly cropped to the size of 160 × 192× 128. Due to the computational limitations, we resize the images from 160 × 192 × 128 to 80 × 96 × 64. For every input image, we randomly flip it across a random axis.

For consideration of robustness, we repeat the process of randomly flipping the input image 10 times for each case and use the flipped data as input to generate 10 training datasets. Using each of these training datasets, we train a model. Finally, we average the results from the 10 models to obtain a final segmentation.

4.2 Results

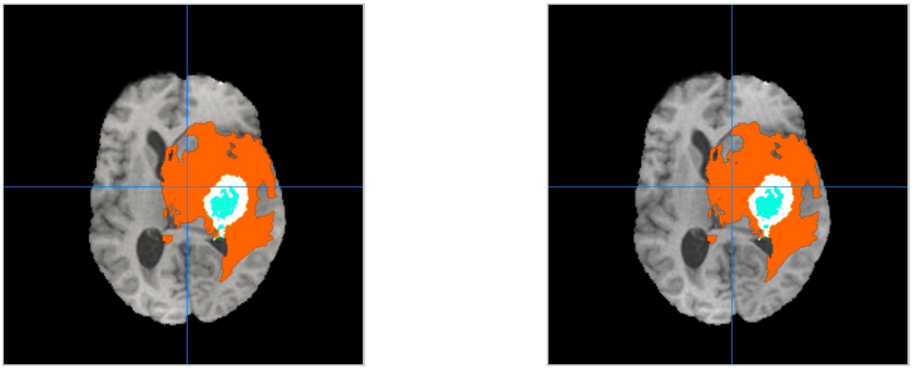
Our model is trained on NVIDIA Tesla V100 16 GB GPUs. If running on a single GPU, the approximate time for training 300 epochs is about 2.5 days. The best result of a single model on the validation dataset is provided in Table 2. The best final result on the validation dataset is reported in Table 3, which is obtained by averaging the results from 10 models trained on the 10 training datasets. The model ensemble improves our results by 0.4%. An example of our predicted segmentation is shown in Fig. 5.

Table 2. BraTS 2020 validation dataset result of the best single model.

	Dice			Hausdorff (mm)		
Validation	ET	WT	TC	ET	WT	TC
Best model	0.69800	0.88934	0.78357	34.29461	4.5323	10.06072

Table 3. BraTS 2020 validation dataset result of the best ensembled model.

	Dice			Hausdorff (mm)		
Validation	ET	WT	TC	ET	WT	TC
Best model	0.70301	0.89292	0.78977	34.30576	4.6287	10.07086



(a) Predicted

(b) Original

Fig. 5. An example of our predicted segmentation. The blue area represents the Necrotic and Non-Enhancing Tumor (NCR/NET), the orange area represents the peritumoral edema (ED) and the white area represents the GD-enhancing tumor (ET) (Color figure online)

5 Discussion and Conclusion

In this paper, we modified a VAE-regularized 3D U-Net to the new proposed MultiResUNet architecture by replacing the classic ResNet with the MultiRes block and adding Res Path between the encoder and decoder parts to reduce the possible semantic gap. The MultiRes Block extracts more information on different scales of the image. We experimented our method on the BraTS 2020 training dataset and validation dataset. The result has shown that the architecture has satisfactory performance. We have tried to train the 3D U-Net model proposed by the top-1 winner Myronenko [10] in BraTS 2018 Challenge. However, we met a challenge of limited computational resource since Myronenko trained his model on the NVIDIA Volta V100 32 GB GPU. We tried to crop the original image from 240 × 240 × 150 to 112 × 126 × 96 which may largely affect the performance of that model. With the increasing amount of data, the model would expect to have higher requirements for GPUs. In comparison, the result of our model is very close to the result of that model but requires less computational resource, which makes our model more economically attractive.

References

1. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283. USENIX Association, Savannah, GA, November 2016. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi
2. Bakas, S., et al.: Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features. Sci. Data **4** (2017). https://doi.org/10.1038/sdata.2017.117
3. Bakas, S., et al.: Segmentation labels and radiomic features for the pre-operative scans of the TCGA-LGG collection, July 2017. https://doi.org/10.7937/K9/TCIA.2017.GJQ7R0EF
4. Bakas, S., et al.: Segmentation labels and radiomic features for the pre-operative scans of the TCGA-GBM collection, July 2017. https://doi.org/10.7937/K9/TCIA.2017.KLXWJJ1Q
5. Bakas, S., 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). http://arxiv.org/abs/1811.02629
6. Chollet, F., et al.: Keras (2015). https://github.com/fchollet/keras
7. Ibtehaz, N., Rahman, M.S.: Multiresunet: rethinking the u-net architecture for multimodal biomedical image segmentation. Neural Netw. Official J. Int. Neural Netw. Soc. **121**, 74–87 (2020)
8. Jiang, Z., Ding, C., Liu, M., Tao, D.: Two-stage cascaded U-net: 1st place solution to BraTS challenge 2019 segmentation task. In: Crimi, A., Bakas, S. (eds.) BrainLes 2019. LNCS, vol. 11992, pp. 231–241. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-46640-4_22
9. Menze, B.H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., et al.: The multimodal brain tumor image segmentation benchmark (BRATS). IEEE Trans. Med. Imaging **34**(10), 1993–2024 (2015). https://doi.org/10.1109/TMI.2014.2377694
10. Myronenko, A.: 3D MRI brain tumor segmentation using autoencoder regularization. CoRR abs/1810.11654 (2018). http://arxiv.org/abs/1810.11654
11. Naik, S., Doyle, S., Agner, S., Madabhushi, A., Feldman, M., Tomaszewski, J.: Automated gland and nuclei segmentation for grading prostate and breast cancer histopathology, pp. 284–287, June 2008. https://doi.org/10.1109/ISBI.2008.4540988
12. Qayyum, A., Anwar, S.M., Majid, M., Awais, M., Alnowami, M.R.: Medical image analysis using convolutional neural networks: A review. CoRR abs/1709.02250 (2017). http://arxiv.org/abs/1709.02250
13. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28

14. Schindelin, J., Rueden, C., Hiner, M., Eliceiri, K.: The ImageJ ecosystem: an open platform for biomedical image analysis. Mol. Reprod. Dev. **82** (2015). https://doi.org/10.1002/mrd.22489
15. Van Rossum, G., Drake, F.L.: Python 3 Reference Manual. CreateSpace, Scotts Valley, CA (2009)