# Bag of Tricks for 3D MRI Brain Tumor Segmentation

Yuan-Xing Zhao[1,2(✉)], Yan-Ming Zhang[1], and Cheng-Lin Liu[1,2,3]

[1] NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing, China
{yuanxing.zhao, ymzhang, liucl}@nlpr.ia.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China
[3] CAS Center for Excellence of Brain Science and Intelligence Technology, Beijing, China

**Abstract.** 3D brain tumor segmentation is essential for the diagnosis, monitoring, and treatment planning of brain diseases. In recent studies, the Deep Convolution Neural Network (DCNN) is one of the most potent methods for medical image segmentation. In this paper, we review the different kinds of tricks applied to 3D brain tumor segmentation with DNN. We divide such tricks into three main categories: data processing methods including data sampling, random patch-size training, and semi-supervised learning, model devising methods including architecture devising and result fusing, and optimizing processes including warming-up learning and multi-task learning. Most of these approaches are not particular to brain tumor segmentation, but applicable to other medical image segmentation problems as well. Evaluated on the BraTS2019 online testing set, we obtain Dice scores of 0.810, 0.883 and 0.861, and Hausdorff Distances (95th percentile) of 2.447, 4.792, and 5.581 for enhanced tumor core, whole tumor, and tumor core, respectively. Our method won the second place of the BraTS 2019 Challenge for the tumor segmentation.

**Keywords:** Brain tumor segmentation · Deep neural network

## 1 Introduction

Gliomas are the most common primary brain malignancies, with different degrees of aggressiveness. 3D brain tumor segmentation plays a vital role in addressing the diagnosis, monitoring, and treatment planning of brain diseases. Although Deep Convolutional Neural Networks (DCNN) have shown great success in solving general computer vision problems, when applied to MRI image segmentation, they face two special challenges.

First, annotated MRI images are very scarce due to privacy concerns and the high cost of human annotation. For example, the training set of BraTS2019 contains only 355 annotated cases. Given that DCNN typically has millions of parameters, the scarcity of annotated data can hardly guarantee the generalization performance of DCNN. Second, the training of DCNN consumes large GPU memory, while the large volume of 3D MRI image data makes the training of DCNN with large patches and large batches impossible. As a result, by training with small MRI patches, DCNN


A. Crimi and S. Bakas (Eds.): BrainLes 2019, LNCS 11992, pp. 210–220, 2020.
https://doi.org/10.1007/978-3-030-46640-4_20

cannot gain enough receptive fields to model the global structure of brains and the spatial relations between different anatomical regions. Besides the specific difficulties of 3D MRI segmentation, the phenomenon of class imbalance is also a common problem that must be faced in semantic segmentation. Most of the tricks described in this paper are aimed at conquering the problems described above.

Since the introduction of U-Net [1] in 2015, DCNN has become the dominating approach for medical image segmentation. Various new approaches have been proposed based on the original U-Net. Myronenko, A. [2] uses auto-encoder to reconstruct the input image itself and regularize the optimizing process. Isensee, F. [3] takes advantage of other labeled data, using a co-training method. McKinley, R. [4] proposes label-uncertainty loss to models to label noise and uncertainty. These methods were shown to be effective in improving the segmentation, yet further improvements can be achieved by considering and combining various strategies in data processing, network architecture, and learning-algorithm design. In this work, we introduce several useful tricks in model learning and combine them to boost the overall accuracy of the model.

The rest of this paper is organized as follows. Section 2 introduces tricks used in 3D MRI brain-tumor segmentation. Section 3 presents the implementation details and experimental results, and Sect. 4 provides concluding remarks.

## 2 Methods

Many tricks in general DCNN design and training for the image can also be applied to the 3D brain-image segmentation. We divide such tricks into three categories: data processing methods, model designing methods, and optimizing methods.

### 2.1 Data Processing Methods

**Sampling.** Data imbalance has always been a hot topic for segmentation. Commonly, that training data contains an overwhelming number of background voxels, most of which are easy for the classifier to predict, and only a few are difficult. Here, we use two methods to cope with this problem.

*Heuristic Sampling.* To reduce the effect of background voxels, we use a heuristic sampling method to select more informative patches. More concretely, several patches are randomly cropped from the input MRI, and the one that contains the most foreground voxels is selected to feed the model.

*Hard Sample Mining.* A standard solution was known as hard sample mining. At every iteration, the voxels with the largest loss values are selected as training voxels, and their gradients are back-propagated to update the model's parameters. The gradients of other voxels are discarded directly. The percentage of the selected voxels is set to and decreases as the number of iteration increases.

The method described above can be seen as a method that uses a hard threshold to select the difficult sample. Lin, T.Y. [5] proposed a new loss called focal loss to

conquer the class imbalance problem through a soft threshold and outperforms the alternatives of training with the hard sample mining. The loss is defined as (2).

$$p_t = \begin{cases} p & if \quad y = 1 \\ 1 - p & otherwise \end{cases} \tag{1}$$

$$CE(p_t) = -(1 - p_t)^{\gamma} \log(p_t) \tag{2}$$

In the above, $y \in \{\pm 1\}$ specifies the ground-truth class, and $p \in [0, 1]$ is the model's estimated probability for the class with $y = 1$. $\gamma > 0$ is a tunable focusing parameter. When an example is misclassified and $p_t$ is small, the modulating factor is near 1, and the loss is unaffected. As $p_t \to 1$, the factor goes to 0, and the loss for well-classified examples is down-weighted. The focusing parameter $\gamma$ smoothly adjusts the rate at which easy examples are down-weighted. Experiment results demonstrate that focal loss is better than general hard sample mining.

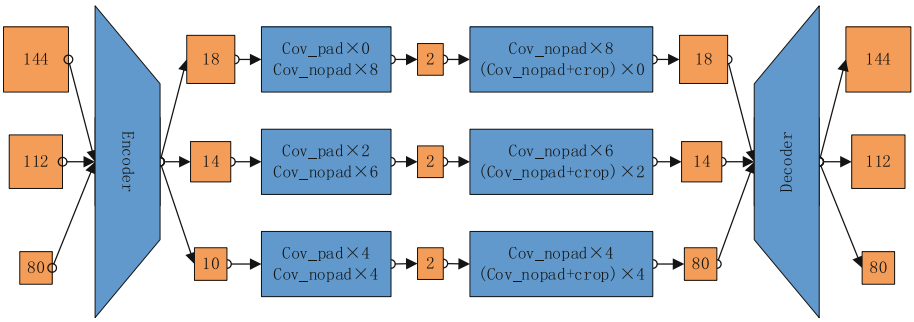


**Fig. 1.** Random patch-size training strategy. At each iteration, we select a training batch with random patch size and using padding and cropping to adjust the size of the feature map. The number in each square is the size of the patch.

**Random Patch-Size Training.** Using large patches could include more contextual information but leads to the small batch size, which increases the variance of stochastic gradients and hurts optimization. On the other hand, using a large batch size could facilitate the optimization but leads to small patches, which results in less contextual information. To take benefit of both sizes of patches, we construct a training batch pool with batches of different patch sizes. Note that if the size of the patch is large, the corresponding batch size will be small.

This method is illustrated in Fig. 1. Using the different numbers of padding and cropping layers between the convolution layer, this model can learn global information from the largest patch and informative texture from the small patch with the same parameter. For each iteration during training, we randomly select a batch from the pool to update the model. We take advantage of both the large patches and the large batch size. In practice, we found this simple strategy very efficient.

**Semi-supervised Learning.** To tackle the lack of annotated data, we use a semi-supervised method called the multi-space semi-supervised method. At the first iteration, the manually labeled dataset is used as a training set, *and* different student-models, $s_i^0$ are trained on the training set under some different conditions, such as the different subset of training set or the different subspace of features, etc. Then, all student-models are combined as a teacher model, so that the teacher model is defined as

$$T^0 = \frac{1}{n}\sum\nolimits_{i=1}^{n} s_i^0 \tag{3}$$

Finally, the teacher model, $T^0$, is used to label the unlabeled dataset.

After the first iteration, we combine the manually labeled dataset and model labeled dataset as the new training set and then repeat the training process as the first iteration. We repeated the process until the accuracy of the student model is stable. Such a process is summarized in Fig. 2.

**Algorithm 1**: Multi-View Semi-Supervised Segmentation Algorithm

1. **Input:**
   a. Overlapped MRI labeled patches $\{(x_l, y_l)\}$ and unlabeled patches $\{x_u\}$
   b. Data transformation functions: $\{(T_v, T_v^{-1})\}$
2. **Training the ensemble model:**
   a. Initialize pseudo-labels $\{y_u\}$ by some supervised, trained model
   b. Repeat
      - For each transformation function $T_v$, update the model $f_v$:
      $$f_v = \underset{f}{\operatorname{argmin}} \sum_{l} L(T_v(x_l), T_v(y_l), f) + \alpha \sum_{u} L(T_v(x_u), T_v(y_u), f)$$
      - Update the ensemble model by Eq. (4)
      - Update pseudo labels $\{y_u\}$ by the method described in Sec.2.2

      Until convergence
3. **Training the student model:**
   a. Learn a student model based on $F$, $\{(x_l, y_l)\}$, $\{x_u\}$

**Fig. 2.** Multi-view semi-supervised segmentation algorithm

### 2.2 Model Devising Methods

**Architecture Devising.** The whole architecture used in this paper is shown in Fig. 3. For many computer vision tasks, a classic way to boost the accuracy is to combine multiple prediction results made at different scales. Inspired by this observation, we introduce a new architecture, named self-ensemble, that makes predictions at each scale of U-Net and then joins them to obtain the final prediction. The simple way to combine

predictions of different sizes is to resize them to the size of the input image, as illustrated on the left side of Fig. 4. However, it is highly memory consuming since it up-samples every prediction tensor to the largest size. Instead, we propose to combine the predictions in a recursive manner which is illustrated at the right side of Fig. 4 and formulated as

$$y_s = \mathrm{Up}(y_{s+1}, 2) + \widetilde{y}_s \qquad s = S-1, S-2, \ldots, 1 \tag{4}$$

$$y = y_1 \tag{5}$$

where $\widetilde{y}_s$ is the prediction tensor of the $s$-th scale, $S$ is the number of scales in U-Net, and $\mathrm{Up}(y, t)$ is a function that up-samples $y$ by rate $t$. The prediction of the current scale $y_s$ is based on $y_{s+1}$ and only needs to model the residual of $y_{s+1}$. The final result of $y = y_1$, combining predictions at each scale, outperforms any single prediction.

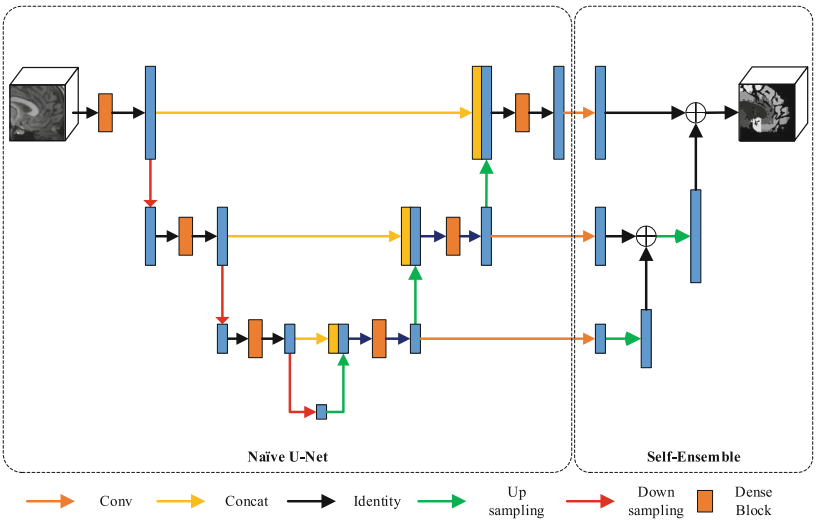


**Fig. 3.** The overall architecture of the self-ensemble U-Net model.

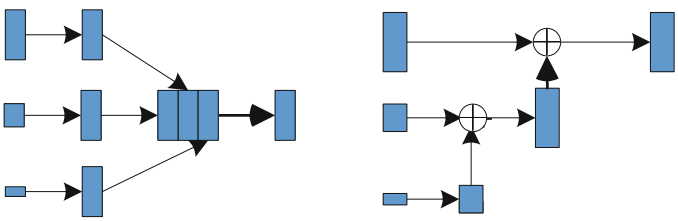

**Fig. 4.** The left figure is the naïve way to combine predictions of different scales. The right figure is our self-ensemble method.

**Fusing Result.** We use two methods for combining different results to improve the final prediction. The overall architecture is shown in Fig. 5. The ensemble model fusing the prediction of different models is shown at the top of Fig. 5, and the method of fusing the prediction of overlapped patches is illustrated at the bottom of Fig. 5.

*Fusing the Prediction of Different Models.* The method combined different models like [3]. We evaluate our model by running five-folds cross-validation on the training cases. Then we use the average of all the five models as the final ensemble model.

*Fusing the Prediction of the Overlapped Patch.* The model may predict the different results of the same voxel because of the voxel located in a different position related to the different patches. Base on this phenomenon, we crop the input MRI into overlapped patches and then combine these patches as a batch predicted by the model. In this way, the overlapped voxel is predicted more than one time. A more accurate result would be obtained by averaging these predictions.

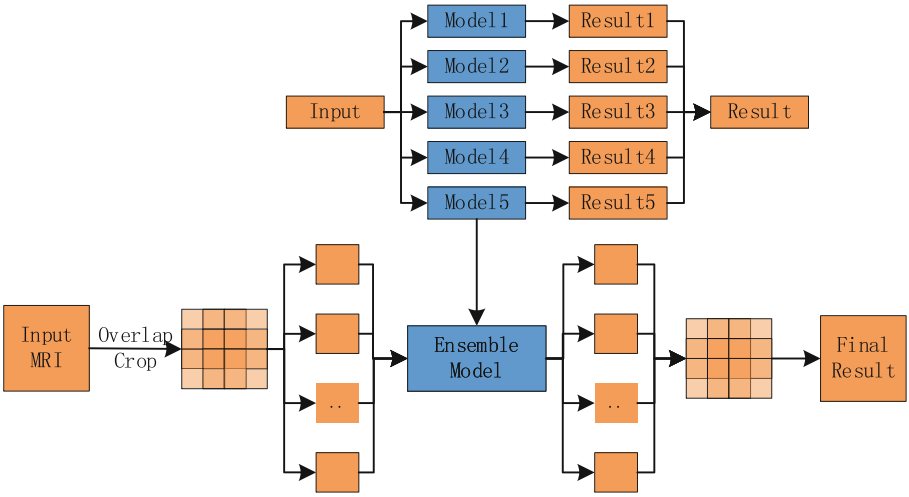


**Fig. 5.** The overall architecture of the result is fusing. At the top of the figure is the method for fusing the predictions of the different models, and at the bottom of the figure is the method fusing the prediction of overlapped patches.

## 2.3 Optimizing Methods

**Gradual Warming Up Learning Rate.** The gradual warming up learning rate, which was first proposed in [6], gradually increases the learning rate from a small value to a large value. In practice, with a mini-batch of size, we start from a learning rate of $\eta$ and increase it by a constant amount at each iteration until it reaches $\widehat{\eta} = k\eta$ after several epochs. After the warmup phase, we go back to the original learning rate schedule.

**Multitask Learning.** Multitask learning can be seen as a regulation method. It can affect the process of optimizing and provide additional information related to the learning problem. In this paper, we first calculate the cross-entropy loss between the softmax result and the label provided by the organizer, including the background (BG), the necrotic and the non-enhancing tumor core (NCR/NET), the peritumoral edema (ED), and the enhancing tumor (ET). Then the predicted result and ground truth are reorganized as four independent categories. These include background, enhancing tumor, whole tumor (WT), and tumor core (TC). Finally, the binary cross-entropy loss is calculated between the reorganized prediction and ground truth. The overall process can be seen as Fig. 6.

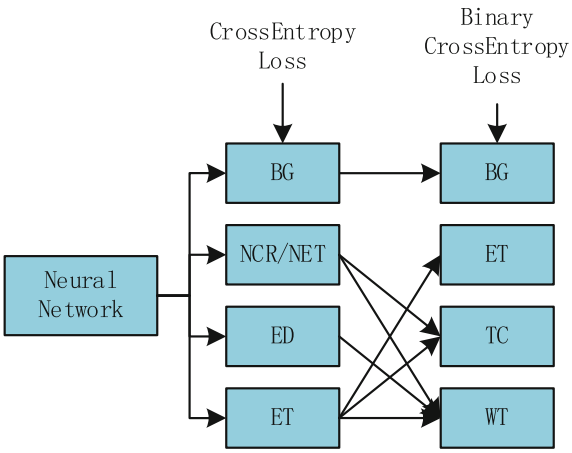


**Fig. 6.** Multitask learning. Optimizing the cross-entropy loss and the binary cross-entropy loss simultaneously.

## 3 Experiments and Results

### 3.1 Datasets and Evaluation Metrics

**Datasets.** We use two datasets in our experiments.

*BraTS2019 [7–11].* It contains 355 cases whose corresponding manual segmentation is provided. Each case has four MRI sequences that are named T1, T1 contrast-enhanced, T2, and FLAIR, respectively.

*Decathlon [12].* It comprises 750 cases collected from older BraTS challenges. We use this dataset as the unlabeled dataset.

**Evaluation Metrics.** The segmentation performance was quantitatively assessed using the mean Dice coefficient (DSC). Let A and B denote the manual label and predicted label, respectively. The mean Dice similarity coefficient is defined as

$$DSC = \frac{1}{n}\sum_{i=1}^{n}\frac{2|A_iB_i|}{|A_i| + |B_i|} \tag{6}$$

where $|A_i|$ denotes the number of positive elements in the binary segmentation $A_i$, and $|A_iB_i|$ is the number of positive elements shared by $A_i$ and $B_i$. n = 4 is the number of labels.

### 3.2 Preprocessing

In our approach, before feeding the data to the deep neural network, each MRI sequence of a case is normalized independently. Specifically, all voxels of an MRI sequence are normalized to range from 0 to 1. We also apply a random axis mirror along the horizontal axis.

### 3.3 Implementation Details

We use the summation of cross-entropy and average Dice similarity as the loss function. The patch size is randomly selected from 64, 80, 96, 112, 128, 144, and the corresponding batch size is 15, 8, 4, 2, 1, 1. SGD with momentum is used as the optimizer, and the learning rate is set to 0.4. The step of warming up is set to 20 epochs. The model is trained in an end-to-end way, and no additional preprocessing or post-processing is performed. The method is implemented by Pytorch, and all experiments are conducted on two TITAN GPUs with 12G RAM. It took around 21 h to train the model.

### 3.4 Results

To better understand our method, we conduct ablation experiments to examine how some trick affects the final performance. We evaluate the performance under different experimental settings: (1) BL, the original U-Net: equipped with dense block structure and self-ensemble structure. (2) BL+warmup: model 1 with warming-up learning rate. (3) BL+warmup+fuse: model 2 with the resultant fusing of five different models trained by fivefold cross-validation. (4) BL+warmup+fuse+semi: model 3 with semi-supervised learning. From the results listed in Table 1, we can observe that our method steadily improves accuracy with the addition of each component. Three examples are visualized in Fig. 7. We can see that most voxels are segmented correctly, while some errors occur in the small regions and boundaries between regions. Table 2 shows the results of our model on the BraTS 2019 testing dataset.

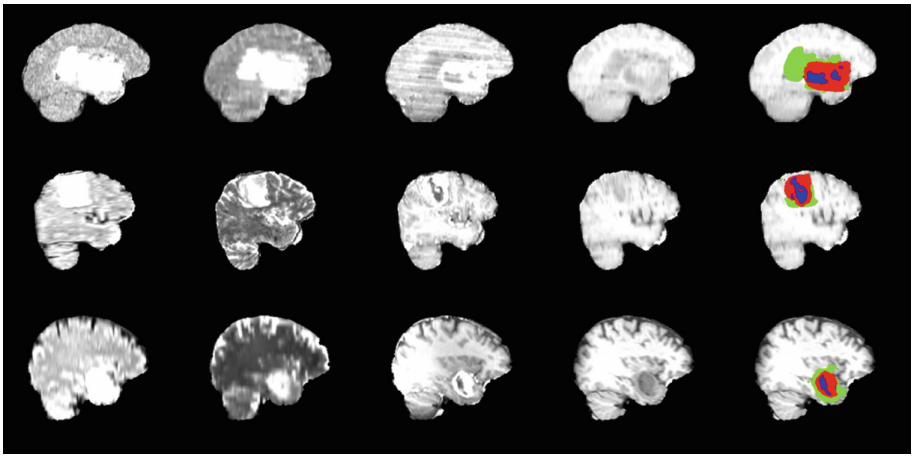

**Fig. 7.** The visualization result of the validation set of the BraTS2019 dataset. From left to right, the column is the original FLAIR image, the original T2 image, the original T1ce image, the original T1 image, and the segmentation result overlaid over the T1 image.

**Table 1.** Results of the BraTS2019 validation data (125 cases). Metrics are computed by the online evaluation platform.

| Method | Dice | | | Hausdorff Dist. | | |
|---|---|---|---|---|---|---|
| | ET | WT | TC | ET | WT | TC |
| BL | 0.702 | 0.893 | 0.800 | 4.766 | 5.078 | 6.472 |
| BL+warmup | 0.729 | 0.904 | 0.802 | 3.832 | 4.141 | 8.099 |
| BL+warmup+fuse | 0.737 | 0.908 | 0.823 | 4.089 | 4.599 | 6.433 |
| BL+warmup+fuse+psudo label | 0.754 | 0.910 | 0.835 | 3.844 | 4.569 | 5.581 |

**Table 2.** Results of the BraTS2019 testing data (166 cases). Metrics are computed by the online evaluation platform.

| | Dice | | | Hausdorff Dist. | | |
|---|---|---|---|---|---|---|
| | ET | WT | TC | ET | WT | TC |
| Mean | 0.810 | 0.883 | 0.861 | 2.447 | 4.792 | 4.217 |
| StdDev | 0.193 | 0.145 | 0.225 | 4.030 | 6.619 | 7.503 |
| Median | 0.850 | 0.924 | 0.928 | 1.732 | 3.0 | 2.236 |
| 25quantile | 0.783 | 0.875 | 0.882 | 1.0 | 1.494 | 1.414 |
| 75quantile | 0.915 | 0.951 | 0.960 | 2.236 | 4.899 | 3.606 |

## 4 Conclusion

In this paper, we review useful tricks for training DCNN to improve the accuracy of brain tumor segmentation and evaluate their performance. Our empirical results on the BraTS2019 indicate that these tricks improve model accuracy consistently. In particular, stacking all of them together leads to significantly higher accuracy. On the BraTS2019 online validation set, our combined method achieved average Dice scores of 0.754, 0.910, 0.835 for the enhancing tumor, whole tumor, and tumor core, respectively. However, our model tends to make false predictions for small anatomical regions.

In the future, we will investigate methods for accurately segmenting small regions and apply them to other tasks such as prediction of patient overall survival from pre-operative scans.

**Acknowledgments.** This work is supported by the National Natural Science Foundation of China(NSFC) Grants 61773376, 61721004, 61836014, as well as Beijing Science and Technology Program Grant Z181100008918010.

## References

1. 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
2. Myronenko, A.: 3D MRI brain tumor segmentation using autoencoder regularization. In: Crimi, A., Bakas, S., Kuijf, H., Keyvan, F., Reyes, M., van Walsum, T. (eds.) BrainLes 2018. LNCS, vol. 11384, pp. 311–320. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11726-9_28
3. Isensee, F., Kickingereder, P., Wick, W., Bendszus, M., Maier-Hein, Klaus H.: No new-net. In: Crimi, A., Bakas, S., Kuijf, H., Keyvan, F., Reyes, M., van Walsum, T. (eds.) BrainLes 2018. LNCS, vol. 11384, pp. 234–244. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11726-9_21
4. McKinley, R., Meier, R., Wiest, R.: Ensembles of densely-connected CNNs with label-uncertainty for brain tumor segmentation. In: Crimi, A., Bakas, S., Kuijf, H., Keyvan, F., Reyes, M., van Walsum, T. (eds.) BrainLes 2018. LNCS, vol. 11384, pp. 456–465. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11726-9_40
5. Lin, T.-Y., et al.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
6. Goyal, P., et al.: Accurate, large minibatch SGD: Training imagenet in 1 h. arXiv preprint arXiv:1706.02677 (2017)
7. Bakas, S., et al.: Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge. arXiv preprint arXiv:1811.02629 (2018)
8. Menze, B.H., et al.: The multimodal brain tumor image segmentation benchmark (BRATS) **34**, 1993–2024 (2014). https://doi.org/10.1109/tmi.2014.2377694

9. Bakas, S., et al.: Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features. **4**, 170117 (2017). https://doi.org/10.1038/sdata.2017.117
10. Bakas, S., et al.: Segmentation labels and radiomic features for the pre-operative scans of the TCGA-GBM collection. **286** (2017). https://doi.org/10.7937/k9/tcia.2017.klxwjj1q
11. Bakas, S., et al.: Segmentation labels and radiomic features for the pre-operative scans of the TCGA-LGG collection. **286** (2017). https://doi.org/10.7937/k9/tcia.2017.gjq7r0ef
12. Simpson, A.L., et al.: A large annotated medical image dataset for the development and evaluation of segmentation algorithms. arXiv preprint arXiv:1902.09063 (2019)