Grenze International Journal of Engineering and Technology, Jan Issue

Deep Learning based Model for Brain Tumor Detection and Segmentation using BRATS Dataset

Kunal Bhujbal[1], Dr. Pradip Mane[2] and Ananya More[3]
[1, 3]Undergraduate student, Department of Computer Engineering, Vasantdada Patil Pratishthan's College of Engineering & Visual Arts, Mumbai, India
Email : kunalbhujbal41035@gmail.com
[2]Associate professor, Department of IT Engineering, Vasantdada Patil Pratishthan's College of Engineering & Visual Arts, Mumbai, India.
Email : pradip.mane@pvppcoe.ac.in

***Abstract*—This paper presents a method that can automatically perform brain tumor segmentation using Deep Neural Networks (DNNs). Glioblastomas (both low and high grade) pictured in MR images are used to tailor the proposed system. These tumors are varied in shape, size and contrast and can be found anywhere in the brain. Our exploration of an extremely efficient machine learning solution run on a flexible and high capacity DNN is inspired by this nature of the tumors. We have worked on various choices of models indoor to promise a competitive performance. The different architectures that we explored were based on Convolutional Neural Networks (CNN). These architectures are also called DNN and their key feature is their adaptability to image data.**

Our novel CNN architecture is different from the traditionally used ones in Computer Vision for the following reasons. Both local features and global features are simultaneously exploited. A convolutional implementation of a layer which is fully connected is used as a final layer in our networks. This speeds up the process by a 40 fold. Difficulties which relate to the imbalance of tumor labels are handled by introducing a 2-phase training procedure. In the end, a cascade architecture is explored. Here for a subsequent CNN, the output of a basic CNN is used as an additional source of information.

***Index Terms*— Deep Learning, CNN, Brain Tumor, Machine Learning, Computer Vision, BRATS 2020.**

I. INTRODUCTION

According to an estimation, 23,000 new cases of brain cancer will be diagnosed in the United States alone, in 2020. The most common brain tumors are gliomas. For a patient with a life expectancy of several years, gliomas can be less aggressive (i.e. low grade) while for a patient with a life expectancy of at most 2 years, it can be more aggressive (i.e. high grade).

Some tumors like gliomas and glioblastomas are very difficult to localise, whereas tumors such as meningiomas can be segmentend easily. What makes the tumors difficult to segment is that they often diffused along with their surrounding edema, they are contrasted poorly and have structures like tentacle extended. As discussed earlier, brain tumors of almost any shape and size can appear anywhere in the brain, which also adds up to the difficulty in segmentation. Furthermore, there is no standardised scale of voxel values in MR images, unlike the images derived from X-ray computed tomography (CT) When pictured in different hospitals, the same tumorous cells may end up having gray scale values which drastically different. These values depend on the ype of MR machine used (1.5, 3 or 7 tesla) and the acquisition protocol (field of view value, voxel resolution, gradient

Grenze ID: 01.GIJET.9.1.309

strength, b0 value, etc).

The white matter, the gray matter, and the cerebrospinal fluid are typically the 3 types of tissues that a healthy brain is made up of. Detection of the location and extension of the tumor regions which may include, active tumours tissue (vascularized or not), necrotic tissue, and edema (swelling near the tumor), is the goal of brain tumor segmentation. This is done by identifying abnormal areas when compared to normal tissue. Glioblastomas are infiltrative tumors. It is difficult to distinguish them from healthy tissues and their borders are often fuzzy. To solve this difficulty, employment of more than one MRI modality is often can be done, e.g. T1 (spin-lattice relaxation), T1-contrasted (T1C), T2 (spin-spin relaxation), proton density (PD) contrast imaging, diffusion MRI (dMRI), and fluid attenuation inversion recovery (FLAIR) pulse sequences. A unique signature to each tissue type is almost given to these modalities due to the contrast between them.

Hand designed features are used by most of the automatic brain tumor segmentation methods. [5] [6]. A classical machine learning pipeline is used to implement these methods wherein features provided to a classifier are first extracted. The nature of those features are not affected by the training procedure. The in domain data can be directly used to learn a hierarchy of increasingly complex features which can be an alternative approach for designing task-adapted feature representations. Such feature hierarchies are best learned by the Deep neural networks [14]. Feature hierarchies combine information across MRI modalities and are adapted specifically to the task of brain tumor segmentation are learnt in this approach.

CNNs, which are DNNs adapted to image data are specifically investigated by us. A well established metrics is used to report their performance, advantages and disadvantages. While CNNs have also been successfully applied to segmentation problems [16][17][18][19], most of the previous work has focused on non-medical tasks and many involve architectures that are not well suited to medical imagery or brain tumor segmentation in particular. They are not well suited to medical imagery or brain tumor segmentation in particular. Brain tumor segmentation using convolutional neural networks with two other methods was presented in BRATS workshop, have been our work of focus. However, those results required more investigation as they were incomplete (More on this in Section 2).

In this paper, tackling brain tumor segmentation using a number of specific CNN architectures is proposed. CNN designs and training techniques have seen the most recent advances in CNN design and training techniques, such as Max-out [20] hidden units and Dropout [3] regularization. These are exploited by our architectures. Architectures that take both the local shape of tumors as well as their context into account are investigated by us.

The analysis of our experiment focuses on the fully-annotated MICCAI brain tumor segmentation (BRATS) challenge 2020 data-set [6]. It contains a well defined splits for training and testing data which helps in comparing directly and quantitatively to a wide variety of other methods.

II. Literature Survey

There has been an exponential growth in the number of publications devoted to automated brain tumor segmentation last several decades, as noted by [5]. This observation tells us that research in this area still has a long wait ago. It also lays emphasis on the tools needed for automatic brain tumor segmentation.

There are two brief categories in which we can divide the Brain tumor segmentation methods. One which are based on generative models and another which are based on discriminative models. These are specifically devoted to MRI. [5][22][23].

Few factors such as domain-specific prior knowledge about the appearance of both healthy and tumorous tissues are crucially the factors that the Generative models rely upon. The process of characterising the tissue appearance is a challenging task. In the existing generative models a tumor is usually identified by the shape or a signal which deviates from a normal (or average) brain [24]. Aligning the 3D MR image on an atlas or a template that is computed from several healthy brains generates anatomical models [25]. These models are typically the models that these methods reply upon. A typical generative model of MR brain images can be found in [26]. In this method the brain is aligned to the ICBM brain atlas and posterior probabilities of healthy tissues (white matter, gray matter and cerebrospinal fluid) are computed.Localizing voxels which have a posterior probability below than a certain threshold helps find the tumorous region.

Inorder to ensure good spatial regularity a post-processing step is then applied. Inorder to get a probability map for abnormalities register the brain images onto an atlas, [27]. Initialise an active contour on this map. Until the change in the posterior probability is below a certain threshold, it is iterated.

Numerous other active-contour methods which follow the same lines and depend on the symmetry of the left-right brain features and/or features based on alignment have been proposed. [28][29][30]. Registration and tumor segmentation takes place at the same time in some methods, because it is challenging to align a brain with a large tumor onto a template [31][32].

Discriminative models are employed as other approaches for brain tumor segmentation. These approaches are different from the generative modelling approaches as they exploit little prior knowledge on the brain's anatomy. They mostly rely on the extraction of [a large number of] low level image features. They directly model the connection between these features and the label of a given voxel. These features may be raw input pixels values [33][[34], local histograms [35][36] texture features such as Gabor filterbanks [37][11], or alignment-based features such as inter-image gradient, region shape difference, and symmetry analysis [38]. Classical discriminative learning techniques have also been used, for example, SVMs [21][39][40] and decision forests [41]. The methods relying on random forests are among the most accurate. This is suggested by the results from 2019, 2020 and 2021 editions of the MICCAI-BRATS Challenge [5][42][35].

Implementation of a conventional machine learning pipeline that reply on hand-designed feature are one common aspect of discriminative models. For these methods, the behaviour of the classifier does not depend on the nature of input features. Hence these features have a sufficiently high discriminative power. Assuming the same, the healthy and non healthy tissues are separated by a trained classifier. When these hand-designed features are used by many traditional machine learning techniques, they need a computation of a large number of features in order to ensure accuracy. This is one of the difficulty of these methods, as it make the computation slower and expensive memory wise. Various efficient techniques use dimensionality reduction or feature selection methods indoor to lower the number of features, but this sometimes results in reduced accuracy.

It is the nature of many of the hand-engineered features, without any specific adaptation to the domain of brain tumors, to exploit the edge related information which is very generic. Composed features that are redefined into higher-level, task-adapted representations are ideally liked by one. Using deep CNNs for brain tumor segmentation looks very promising as an approach as suggested by the recent preliminary investigations. (see the BRATS 2014 challenge workshop papers of [43][13][12]. All three methods divide the 3D MR images into [43][13] or 3D patches [12] and train a CNN to predict its center pixel class. Urban et al. (2014) as well as [13] have a fairly common CNN, consisting of a series of convolutional layers, a non-linear activation function between each layer and a soft-max output layer. Our work here extends our preliminary results presented in [43], using a two-pathway architecture, which we use here as a building block. The preliminary results presented in [43] that uses a two-pathway architecture, which we will be using as a building block, are extended by our work.

Typically natural scene labelling is an application on the CNN-based segmentation models, in computer vision. The RGB channels of a patch from a color image are the inputs for these tasks. A basic CNN is used to make predictions for each pixel in the work of [44]. These predictions are then used as an extra information in the input of the second CNN model for further improvisation. Several distinct CNNs processing the image at different resolutions are involved in the other work [45]. Integration of all the information that is learned from all the CNNs helps make the final per-pixel class prediction. A more global super-pixel segmentation of the image is used to regularise these predictions. This helps produce a smooth segmentation. For semantic scene segmentation, convolutional operations are exploited in the final layer of the network to extend traditional CNN architectures [17]. This is similar to our work.

In general, the work using CNNs for segmentation is comparatively less In the medical imaging domain. However, prediction of the boundaries of neutral tissue in electron microscopy images by using CNNs is some of the recent and notable work by [46]. Here we have made an attempt to explore all the various approaches with their similarities and put forth ours, but in the context of brain tumor segmentation.

III. Our Approach based on Convolutional Neural Network

The BRATS dataset has two dimensional images of the human brain. Segmentation is performed slice by slice from the axial view. Thus, each 2D axial image is processed sequentially where all the pixels are associated with different image modalities. They are as follows : T1, T2, T1C and FLAIR. The method proposed processes the $M X M$ patch centered on the pixel to predict its class.

Convolutional layer is the main building block used to construct CNN architecture. Several layers are stacked on top of each other forming a hierarchy of features. Each layer extracts features from the preceding layer to the hierarchy. A single convolutional layer takes a stack of input planes as an input and produces a few number of

output planes or feature maps as an output. Each feature can be visualized as a topologically arranged map of non linear feature extractor responses applied identically in sliding window fashion to each spatial neighborhood of input planes. For the first convolutional layer, individual input planes are associated with different MRI modalities. The input planes in subsequent layers typically consist of the feature map of the previous layer.
In convolutional layer, computing a feature map consists of the following three steps :

A. Convolution of Kernels

Each feature map Os is either associated with one kernel or several kernels. Computation of feature map Os is shown below.

$$O_s = b_s + \sum_r \quad W_{sr} * X_r \qquad (1)$$

X_r = rth input channel
W_{sr} = Sub kernel for the channel
* = Convolution Operation
b_s = bias term
It means, the affine operation which is performed for each feature map is the addition of the application of R different 2D N X N convolutional filters and a bias term that is added to each resulting spatial position pixel wise. However, an input to the operation is a 3D tensor M x M x R but the spatial topology considered in the XY axial plane is 2D. The key feature of convolutional neural networks is their ability to learn the biases and weights of individual feature maps leading to customized, data driven and task specific dense feature extractors. These parameters are adopted on a surrogate loss function via stochastic gradient descent related to the misclassification error, with gradients figured efficiently via the algorithm for back-propagation.
Special attention is paid to the border pixel treatment by the convolution. Throughout the architecture, for the positions of the pixels that are away from the image border less than N/2 pixels, filter response is not computed. The Q x Q output is a result of the N x N filter convoluted with the patch of M x M input.
Q = M - N + 1
As per Fig. 1, M = 7, N = 3 and Q = 5. Users must specify the hyper-parameters such as size of kernel.

B. Non-linear Activation Function

To the kernel convolution's result, non-linearity is applied elementwise to acquire the features that are non-linear transformation of the input. Max out non linearity proposed by [1], is effective at modeling useful features. Max-out features are related to multiple kernels W_s. It means, each Max-out map Z_s is related with k feature maps : $\{O_s, O_{s+1}, \ldots\ldots, O_{s+K-1}\}$. Max-out maps in Fig. 1 are related with K = 2 feature maps.
Features of max-out related to taking the max over the feature maps O, individually for each spatial location :

$$Z_{s,i,j} = \max \{O_{s,i,j}, O_{s+1,i,j}, \ldots\ldots, O_{s+K-1,i,j}\} \qquad (2)$$

i, j = spatial positions
Max-out features are somehow similar to using convex activation functions whose shape relies on the values of kernels and are adaptive.

C. Max Pooling

It consists of taking the max neuron (feature) value within each feature map over sub-windows. This can be formalized as follows :

$$H_{s,i,j} = max_p \; Z_{s,\; i+p,\; j+p} \qquad (3)$$

p = max pooling window size
Size of the feature map shrinks due to the max pooling operation. It can be controlled by the stride hyper-parameters which are related to the vertical and horizontal increments at which pooling sub-windows are positioned and the pooling size p.
Let,
S = The Stride Value
Q x Q = Shape of the feature map (Before max-pooling)
Size of the output of the max pooling will be D x D. where,

$$D = (Q - p) / S + 1$$

For Fig. 1, the result of max pooling operation is D = 4 output feature map, because Q = 5, p = 2, S = 1. Introduction of invariance to the local translation is the motivation for this operation. Procedure of sub-sampling [2] is found beneficial in other applications.
From the point of view of neural networks, feature maps are related to neurons or the layer of hidden units. In particular, each coordinate in a feature map relates to a singular neuron, for which the size of the receptive field

corresponds to the kernel's size. Manier times it is found that learned kernels indicate edge detectors, each kernel tuned to a spatial frequency, orientation and scale as in proper for the statistics of the training data.

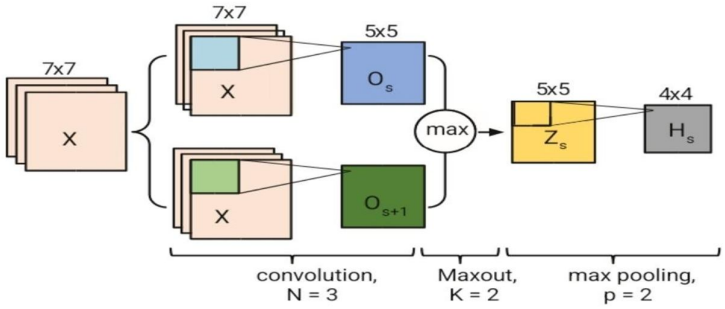

Fig 1. Computation of a single feature map shown by a single convolution layer

In the figure above, 7 x 7 input patch shown is convolved with a series of 3 x 3 kernels followed by max out and max pooling. At last we connect the last hidden convolutional layer to the output convolutional layer followed by a non-linearity to perform a prediction of the segmentation labels. Since the output layer is almost fully connected, a convolutional CNN may not yield an efficient test time for segmentation. The prediction for a whole brain at test time will be 40 times faster by using convolution at the end. The convolution utilizes however many kernels as there are different segmentation labels. Every kernel hence acts as a definitive detector of tissue from one of the segmentation labels. For normalizing the kernel convolution's result into multinominal distribution a softmax nonlinearity is used. Let,

a = Vector of values at the given spatial position

softmax(a) = exp(a) / Z

Where, $Z = \sum_c \quad exp(a_c)$ = Normalization constant

Since over the input patch X, nothing Y as the segmentation label find, thus subsequently we can decipher each spatial position of the convolutional result layer as providing model for the distribution of likelihood $P(Y_{ij} \mid X)$.

Y_{ij} = Label at the position i, j.

Probability of all labels can be calculated by taking product of every conditional $P(Y|X) = \prod_{ij} \quad P(Y_{ij} \mid X)$. A multi class labeling is performed in this approach by assigning the label with largest probability.

D. The Architectures

The most regularly implemented architecture for computer vision work involves a stack of convolutional layers. While composing CNN, various architectures were explored by concatenating feature maps from different layers. This operation helps us to build architecture with multiple computational paths, different purposes can be served by each. Following are the two types of architectures explored in the paper.

Two-Pathway Architecture

Two streams make this architecture : 7 x 7 and 13 x 13 receptive fields pathway. These streams are referred to as local and global pathways respectively. The inspiration for this architectural decision is that we would need the prediction of the pixel's label to be impacted by two factors: Visual details around that pixel and the larger context of it, i.e. generally where the patch is in the brain.

Two layers with 3 x 3 kernel (for the 2nd layer) for the local pathway have been used for the concatenation of the top hidden layer.

Cascaded Architecture

CNNs predict labels of each segmentation separately, it's one of their disadvantages. It is different from segmentation methods which frequently propose joint models of the labels of the segmentation. One of the approaches is to perform a message passing interface of the mean-field by defining Conditional Random Field (CRF) for producing a complete segmentation. For this case, beliefs of the models about the vicinity of the position of the label influences the final label for a given position.

On the contrary, simple feed forward pass down CNN is computationally less expensive than joint segmentation methods. This is a significant viewpoint that one ought to consider if automatic brain tumour segmentation has to be used for day-to-day applications.

Architectures of CNN which expose the efficiency of CNN and also model the dependencies among adjacent labels in the segmentation. Since the need is definitive prediction to be influenced by the

beliefs of the model about the label's value, it has been proposed to feed probabilities of output of the first CNN as supplementary inputs to the second CNN layer. It is done by depending on the concatenation of the convolutional layer. Here, the first CNN's output layer is concatenated with any of the layers of the second CNN. The same two pathway structure has been used for both CNNs.
In this work, three cascaded architectures have been investigated which concatenates the output of first CNN at different levels of the 2nd CNN.

- Input Concatenation : In this architecture, the second CNN directly receives the output of the first CNN. Thus, for the input patch they work as additional image channels. The details are illustrated in Fig. 2A. This model is referred to as INPUTCASCADECNN.
- Local Pathway Concatenation : In this architecture, in the local pathway, we perform concatenation in the second CNN by moving up one layer. Fig. 2B illustrates the details. This model is referred as LOCALCASCADECNN.
- Pre-output Concatenation : In this architecture, we perform concatenation by moving to the very end of the second CNN before its output layer. Computations made by this architecture are similar to the computations made in a CRF by one pass of mean-field inference. From this perspective, the first iteration of the mean field is the output of the second CNN, while the second iteration is the output of the second CNN. Fig. 2C illustrates the details. Output at one position is allowed to be influenced by its previous value in the CNN model presented in this paper and that's the difference from regular mean field. This model is referred to as MFCASCADECNN.

E. Training

Gradient Descent

Through interpretation of the CNN's output over labels of the segmentation for the distribution, criteria of the natural training to boost the probability of all the labels in the training set, or, comparably, to minimize negative log-probability $-\log p(Y|X) = \sum_{ij} -\log p(Y_{ij} \mid X)$ for each brain with a label.
A stochastic gradient descent approach has been followed by selecting Y_{ij} label more than once at arbitrary subset of patches inside each brain, On the CNNs parameters (kernels at all the layers) performing gradient descent step and for the mini batch of patches computing the average negative log probabilities.
For each update, to avoid processing whole brain updates are performed based on only a subset of patches. This approach is implemented by creating a dataset of small batches of image patches of a smaller brain, paired with the subsequent center segmentation label.
To improve optimization further, momentum strategy is implemented which is found to be successful in [2]. For damping optimization velocity a temporally averaged gradient is used.

$V_{i+1} = \mu * V_{i-\alpha} * \nabla W_i$

$W_{i+1} = W_i + V_{i+1}$

Where,

W_i = Parameters of CNN at ith iteration.

∇W_i = At W_i, gradient of the loss function.

V = Integrated Velocity. Initialized at 0.

α = Learning Rate.

μ = Coefficient of momentum

A schedule is defined for where the coefficient of the momentum is increased gradually during the training. Initially, in the experiments performed, $\mu = 0.5$ and finally the value was set to $\mu = 0.9$.
Also, at each epoch, α (the learning rate) is decreased by the factor. Initially, the learning rate was $\alpha = 0.005$. The decay factor is 10^{-1}.

Two Phase Training

Segmentation of Brain tumors can be considered as a problem with highly imbalanced data where the good voxels (i.e. label 0) comprises 97% of the total voxels. Out of the 3% remaining voxels, 0.36% belongs to necrosis (label 1), 2.2% to edema (label 2), 0.15% to non-enhanced (label 3), 0.29% to enhanced tumor (label 4). Choosing patches from the genuine distribution would make the model overwhelmed by the sound patches and causing issues when training out CNN model. All things considered, at first we build patches dataset in such a way that all labels are equiprobable. This was the first training phase. In the second phase, we retrained the output layer with representable distribution of the labels. This is the way we get the best of both worlds.

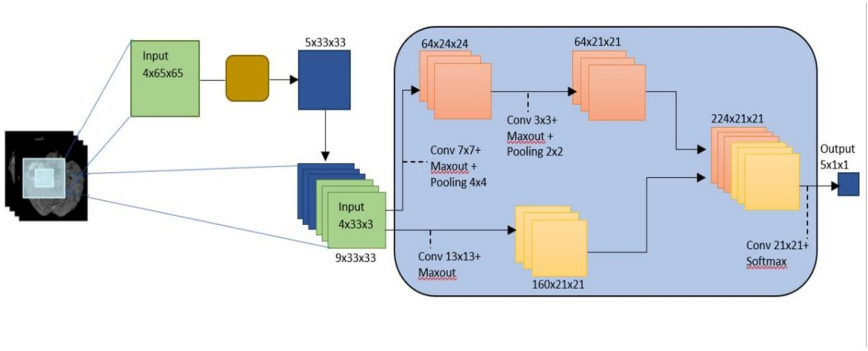

Fig 2A. INPUTCASCADECNN

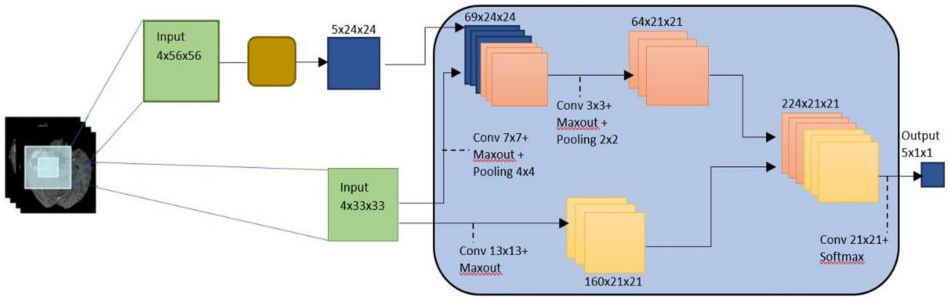

Fig 2B. LOCALCASCADECNN

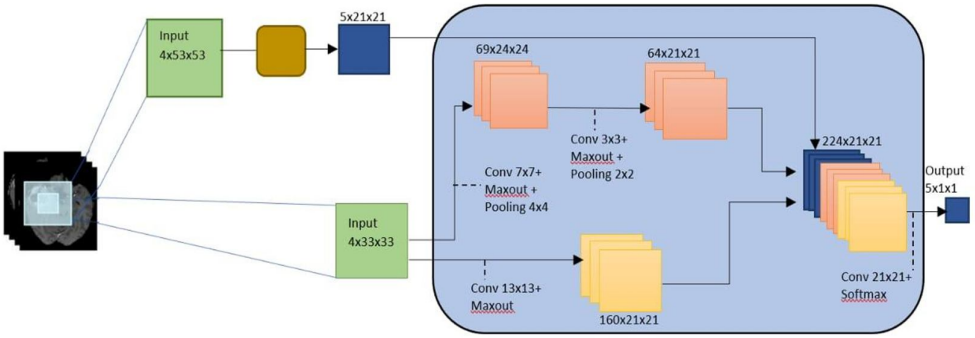

Fig 2C. MFCASCADECNN

Regularization

Successful CNNs will quite often be models with a lot of capacity, making them overfitting vulnerable for situations like us when there aren't enough training examples. Appropriately, we observed that regularization is significant in acquiring good results. Here, regularization takes several forms. In the first layer, for preventing overfitting, we applied both L2 and L1 regularization by bounding the absolute value of kernel weights. It is completed by adding the regularization term to the negative log-probability.

i.e. $-\log p(Y|X) + \lambda_1 ||W||_1 + \lambda_2 ||W||_2$

Where,

λ_2 and λ_1 = coefficients for L2 and L1 regularization terms.

L1 empowers the sparsity while L2 empowers the small values and thus how both L2 and L1 affect parameters of the model. Validation set has been used to stop training when performance of the validation stops improving and also to tune other hyper parameters of the model. The hyper parameters of the model were chosen by performing a great search over the scope of parameters. The hyper parameters which were selected are those for whom the model gave the best performance on a validation set.

Moreover, a Dropout [3] is used, a regularization method which works by adding noise stochastically in the computation of CNN's hidden layer. This is done by masking (multiplying each hidden unit by 0) with a specific

probability, independently for training update and every unit. Since it can't be assumed by each unit that other units aren't masked and they co-adapt themselves, it encourages the network to learn features which are useful on their own. At the test time, units are rather duplicated by one less the likelihood of being masked. As mentioned in [3]. Taking into account the huge number of parameters our model has, one could feel that the 30 training brains with the regularization strategy of us are not sufficient for preventing overfitting. Our model doesn't overfit and generalizes well since it has approximately 5999 2D images to train and each brain with 200 2D slices. Somehow, we cannot distinguish MRI images of different brains because they're very similar from one patient to another. Fewer training samples are needed because the variety in the images of the brain (MRI) is very less as compared to other datasets containing images such as CIFAR and ImageNet.

Cascaded Architecture

For training the cascaded architectures, we start by TWOPATHCNN training with stochastic gradient descent with two phases. By fixing the parameters of the TWOPATHCNN and including it in the cascaded architecture, we move for training the rest parameters using similar procedures. It ought to be seen anyway that there must be a match for the spatial size of the first CNN's result and second CNN's layer; a larger input should be fed to the first layer of CNN. Hence the second CNN should be trained for the larger patches. E.g. As shown in the figure 2A (INPUTCASCADECNN), 65 x 65 is the input size for the first model and the output size is 33 x 33. Only for this case, the concatenation of the first CNN with the second CNN's input channel.

IV. Details of an Implementation

The Pylearn2 library is the backbone of the implementation [4]. Specialized in deep learning algorithms, Pylearn2 is a machine learning library available open source. For increasing the execution speed of deep learning algorithms, Pylearn2 also upholds the use of Graphical Processing unit (GPU).

Only minimal preprocessing is applied on CNNs because they are able to learn useful features from the scratch. Similar preprocessing mentioned in [5] is employed by us. Three steps are followed in the processing. In the first step, intensities in 1% lowest and highest are removed. Later, N4ITK bias correction is applied to T1C and T1 modalities.

Data is then standardized inside every input channel by taking away the channel's mean and dividing by the standard deviation of the channel.

With respect to postprocessing, a straightforward strategy associated with connected components was executed to eliminate flat blobs which could show up in the predictions because of the bright corners of the brains which are close to the skull.

The hyper parameters of the various architectures can be observed in the fig. 2A, 2B and 2C. Hyper parameters were tuned by utilizing cross validation and grid search on the validation set. Hyper parameters which were chosen were the ones on which the model performed supreme on the validation set. Always a stride of 1 was used for max pooling to keep the accuracy of per pixel during the prediction of full image. In the practice, we have observed that max pooling in the global path doesn't contribute to improving accuracy. Likewise, it has also been found that raising the limit of the model by adding extra feature maps to the convolution blocks gives no significant performance improvement.

Biases are instated to zero with the exception of the soft max layer for which they are initialized to the log of label frequencies. The kernels are arbitrarily instated from U(-0.005, 0.005). Training requires around 3 minutes for each epoch of the TwoPathCNN on NVIDIA's Titan Black Card.

In order to completely utilize the computational speed of the GPU, the program is compiled on GPU. In addition, computations get accelerated due to the convolutional nature of the output. By feeding an input as a full image (not individual patches) it is done. Hence, convolutions at each layer can be extended to acquire all label probabilities

$P(Y_{ij} \mid X)$ for an entire image. With this implementation, we are successful in producing segmentation in 25 seconds for each brain with the TwoPathCNN model. It turns out to be 40 times faster as compared to the extraction of patch at every pixel and processing them individually for the entire brain.

Predictions for LOCALCASCADECNN model takes on an average 1.7 minutes while MFCASCADECNN model takes 1.5 minutes and INPUTCASCADECNN takes 3 minutes.

V. Experiments and Results

As part of the MICCAI conference, the experiments used authentic patient records from the brain tumor segmentation challenge (BRATS-2020) [6]. Three sub-data sets constitute the BRATS-2020 dataset. The test data-set, which encompasses 10 (all high-grade cancers), the leader board data-set, which contains 25, and the training data-set, which contains 30 patient subjects with pixel-accurate ground truth (20 high grade and 10 low grade tumors) (21 high grade and 4 low grade tumors). The test and leader board datasets doesn't include any ground truth. The data-set has one orientation for all of the brains. Four co-registered modalities, T1, T1C, T2, and Flair, are co-registered for each brain. The ground truth for the training brains comprises five segmentation labels: non-tumor, necrosis, edema, non-enhancing tumor, and enhancing tumor.

A sample of the data and the actual data are shown in Fig. 3. The model runs through 3.2 million instances of healthy patches and around 2.2 million samples of tumorous patches (which include all 4 sub-tumor classifications). As previously noted, the distribution of instances presented to the model from each of the five classes is uniform throughout the initial phase of training.

Please be aware that we were unable to use the BRATS-2019 data set owing to issues with both the assessment technique and the fineness of the labeled data. We also executed an experiment wherein we trained our model using the aged 2021 data set and made predictions using the test data set, however the results were not as good as those reported in this study. These factors led us to choose to concentrate on the BRATS data. Because the MRI volumes in the data set lack an isotropic resolution and the spacing in the third dimension varies throughout the data, as was outlined in Section 3, we operate with 2D slices. Although we investigated the use of 3D information (by treating the third dimension as additional input channels or by having an architecture that takes orthogonal slices from each view and makes the prediction on the intersecting central pixel), this didn't improve performance and made our method very slow. It should be noted that we used data augmentation by flipping the input frames in accordance with the recommendation made by [2]. The accuracy of our model as a whole did not increase, contrary to what [7] reported.

By submitting the segmentation results to the online BRATS evaluation system, a quantitative analysis of the models' performance on the test set is accomplished [8]. The following quantitative findings are provided by the online system: Three distinct tumor areas include the various tumor structures. This is mostly as a result of real-world therapeutic applications. [5] outlined the following as the definition of tumor regions:

(a) The whole tumor area (including all four tumor structures).

(b) The central area of the tumor, which includes all tumor features other than "edema."

(c) The area surrounding the tumor that is enhancing (including the "enhanced tumor" structure).

Dice (also known as the F measure), Sensitivity, and Specificity are calculated for each tumour location as follows:

$$Dice(P, T) = \frac{P_1 \wedge T_1}{(|P_1| + |T_1|)\ /2},$$

$$Sensitivity(P, T) = \frac{|P_1 \wedge T_1|}{|T_1|},$$

$$Specificity(P, T) = \frac{|P_0 \wedge T_0|}{|T_0|}$$

Where T stands for the ground truth labels and P for the model predictions. We additionally record the subset of voxels that were expected to be positives and negatives for the relevant tumor location as T1 and T0. In the same way, P1 and P0.Every technique presented for review receives a rating from the online evaluation system. This covers both anonymous, unreleased techniques for which no citation is given and methods from the BRATS challenge that were published in [5]. We present research observations for our various CNN designs in this section.

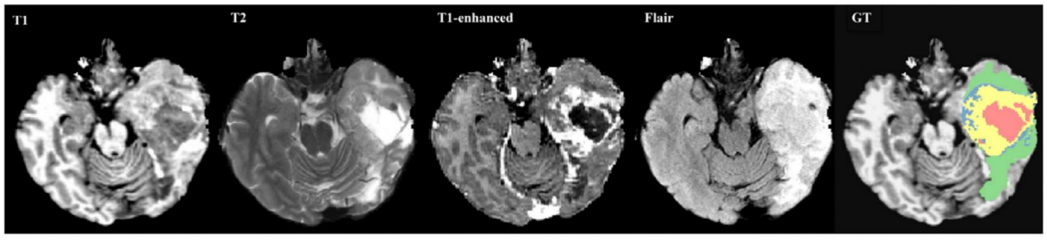

Fig 3. The initial four images from left to right shows the MRI modalities utilized as input channels to different CNN models and fifth image depicts the ground truth labels where edema, necrosis, enhanced tumor, non-enhanced tumor

A. The TWOPATHCNN Architecture

As was previously indicated, unlike traditional CNNs, the TwoPathCNN design encompasses two pathways: a "local" path that is more detail-focused and a "global" path that is more context-focused. We give findings on each pathway as well as results on average the outputs of each pathway when trained independently to better understand how simultaneous training of the global and local pathways enhances performance. Our strategy also tackles the problem's imbalanced character. We present findings with and without the two-phase training to assess its effects. The CNN model with only the local path (i.e., the conventional CNN architecture) is referred to as LocalPathCNN; the CNN model with only the global path is referred to as GlobalPathCNN; the model that averages the outputs of the local and global paths is referred to as AverageCNN; and the two-pathway CNN architecture is referred to as TwoPathCNN. The name of the architecture is marked with the symbol "*" to indicate the second training phase. We only provide results on GlobalPathCNN and AverageCNN with the second phase training because it has a significant impact and always enhances performance.

The quantitative consequences of these modifications are shown in Table 1. Results for the TwoPathCNN with one and two training phases, the common single path CNN (also widely recognized as LocalPathCNN) with one and two training phases, the GlobalPathCNN*, a single path CNN model that adheres to the global pathway architecture, and the output average of each of the trained single-pathway models (AverageCNN*) are included in this table. Unsurprisingly, CNN's single path with a single training phase was placed bottom with the lowest results across nearly all regions. With a rank that increased from 15 to 9, using a second training phase provided the model a big boost. The table also highlights that simultaneous training of the local and global pathways culminates in significantly larger performance than independent training of each pathway accompanied by output averaging. One plausible explanation is that the model enables the two routes to co-adapt by concurrently training the local and global paths. In fact, because the GlobalPathCNN* performs so poorly, the AverageCNN* performs worse than the LocalPathCNN*. The TwoPathCNN*, which has a rank of 4, is the best performing technique in the uncascaded models.

Additionally, sometimes findings for the Enhancing region are less precise than those for the Core and Complete regions. There are primarily two causes for it. First, the boundaries between increased tumor and non-enhanced tissues are often dispersed and imperceptible.

B. Cascaded Architecture

The three cascaded architectures, InputCascadeCNN, LocalCascadeCNN, and MFCascadeCNN, are the subject of our current discussion. The quantitative outcomes for each architecture are shown in Table 2. Additionally, Fig. 4 offers illustrations of the segmentation produced by each architecture.

We discover that the MFCascadeCNN* model produces class borders that are more smoothly defined. We postulate that these parameters are more likely to learn that the center pixel label should be the same as its surroundings because the neurons in the soft-max output layer are directly coupled to the prior outputs within each receptive field.

TABLE 1. TWOPATHCNN MODEL'S PERFORMANCE AND VARIATION. '*' IN FRONT OF THE ARCHITECTURE NAME REPRESENTS SECOND PHASE TRAINING

Rank	Method	Dice			Specificity			Sensitivity		
		Complete	Core	Enhancing	Complete	Core	Enhancing	Complete	Core	Enhancing
4	TWOPATHCNN*	0.85	0.78	0.73	0.93	0.80	0.72	0.80	0.76	0.75
9	LOCALPATHCNN*	0.85	0.74	0.71	0.91	0.75	0.71	0.80	0.77	0.73
10	AVERAGECNN*	0.84	0.75	0.70	0.95	0.83	0.73	0.77	0.74	0.73
14	GLOBALPATHCNN*	0.82	0.73	0.68	0.93	0.81	0.70	0.75	0.65	0.70
14	TWOPATHCNN*	0.78	0.63	0.68	0.67	0.50	0.59	0.96	0.89	0.82
15	LOCALPATHCNN*	0.77	0.64	0.68	0.65	0.52	0.60	0.96	0.87	0.80

Regarding the LocalCascadeCNN* architecture, it reduced false positives in the total tumor category but did not increase performance in the tumor core or enhanced tumor categories.

The segmentation findings from the identical brains (as in Fig. 4) are displayed in Sagittal and Coronal perspectives in Fig. 5. These outcomes were generated using the InputCascadeCNN* model. This image shows that although the segmentation is done in an axial perspective, the results are the same in coronal and sagittal

TABLE II. CASCADED ARCHITECTURE'S PERFORMANCE. RESULTS REPORTED FROM THE SECOND TRAINING PHASE

Rank	Method	Dice			Specificity			Sensitivity		
		Complete	Core	Enhancing	Complete	Core	Enhancing	Complete	Core	Enhancing
2	INPUTCASCADECNN*	0.88	0.79	0.73	0.89	0.79	0.68	0.87	0.79	0.80
4-a	MFCASCADECNN*	0.86	0.77	0.73	0.92	0.80	0.71	0.81	0.76	0.76
4-c	LOCALCASCADECNN*	0.88	0.76	0.72	0.91	0.76	0.70	0.84	0.80	0.75

views. Despite the fact that the segmentation results from the subjects in Fig. 6 is from our validation set, on which the model has not been trained, and that they can provide a good estimate of the model's performance on a test set, we visualize the model's performance on two subjects from the BRATS-2020 testset for greater clarity. In Fig. 7, the Sagittal (top) and Axial (bottom) perspectives of these findings are displayed.

We show in Fig. 6 the advancement of the model by generating predictions on a subject from our validation set every few epochs in order to better understand how InputCascadeCNN* learns features. Let's not forget that, according to [5], Tustison's technique requires 100 minutes to compute predictions for each brain, but InputCascadeCNN* just requires 3 minutes—over 30 times quicker than the challenge winner—thanks to its fully convolutional framework and GPU implementation. The TwoPathCNN* performs in a manner that is nearly cutting edge. However, it is more than 200 times quicker than Tustison's technique, with a forecast time of 25 s. [9] and [10]. are two more top techniques in the chart, with processing durations of 6 and 90 minutes, respectively.

On the BRATS 2020 test dataset, [12] employed an average of two 3D convolutional networks with dice measures of 0.87, 0.77, and 0.73 for Complete, Core, and Enhancing tumor areas. The prediction time was around 1 minute per model, for a total prediction time of 2 minutes. Again, we are unable to fully compare as they do not give Specificity and Sensitivity metrics. However, based on their dice results, our TwoPathCNN* performs similarly while running four times faster—only requiring 25 s. Additionally, the InputCascadeCNN* has the same processing time but is more accurate or equivalent in accuracy. Regarding [13], they omit to include the findings from the BRATS test data set. However, their approach is extremely reminiscent of the LocalPathCNN, which performs worse in our test

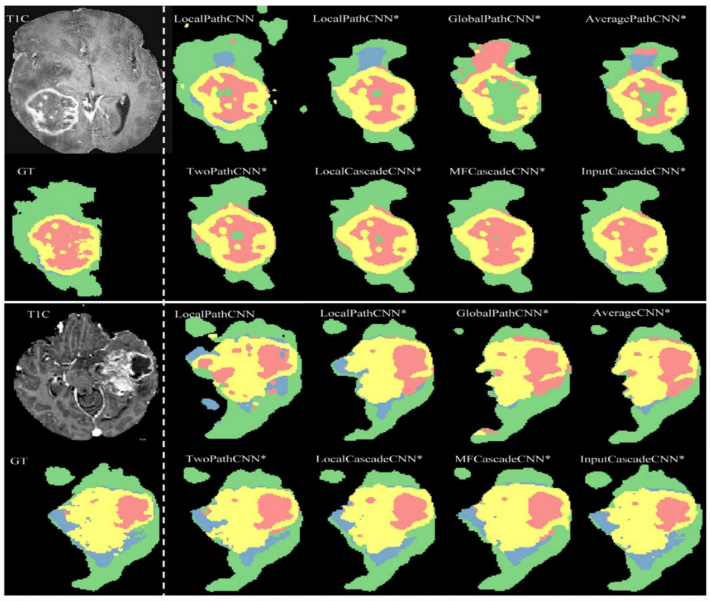

Fig. 4. Axial view of the visual results from CNN architecture. For every sub figure, left to right of a top row depicts T1C modality, conventional one path CNN, conventional CNN with two training phases and TWOPATHCNN model. Ground truth is depicted from left to right of the second row, LOCALCASCADECNN model, MFCASCADECNN model, INPUTCASCADECNN.

Following are the color codes: edema, necrosis, enhanced tumor, non-enhanced tumor

TABLE III. COMPARISON OF THE BEST ARCHITECTURES IMPLEMENTED BY US WITH THE CUTTING EDGE TECHNIQUES ON BRATS 2020 COMPETITORS SET

Method	Dice			Specificity			Sensitivity	
	Complete	Core	Enhancing	Complete	Core	Enhancing	Complete	Core
Enhancing								
INPUTCASCADECNN*	0.84	0.71	0.57	0.88	0.79	0.54	0.84	0.72
0.68								
Tustison	0.79	0.65	0.53	0.83	0.70	0.51	0.81	0.73
0.66								
Zhao	0.79	0.59	0.47	0.77	0.55	0.50	0.85	0.77
0.53								
Meier	0.72	0.60	0.53	0.65	0.62	0.48	0.88	0.69
0.60								
Reza	0.73	0.56	0.51	0.68	0.64	0.48	0.79	0.57
0.63								
Cordier	0.75	0.61	0.46	0.79	0.61	0.43	0.78	0.52
0.52								

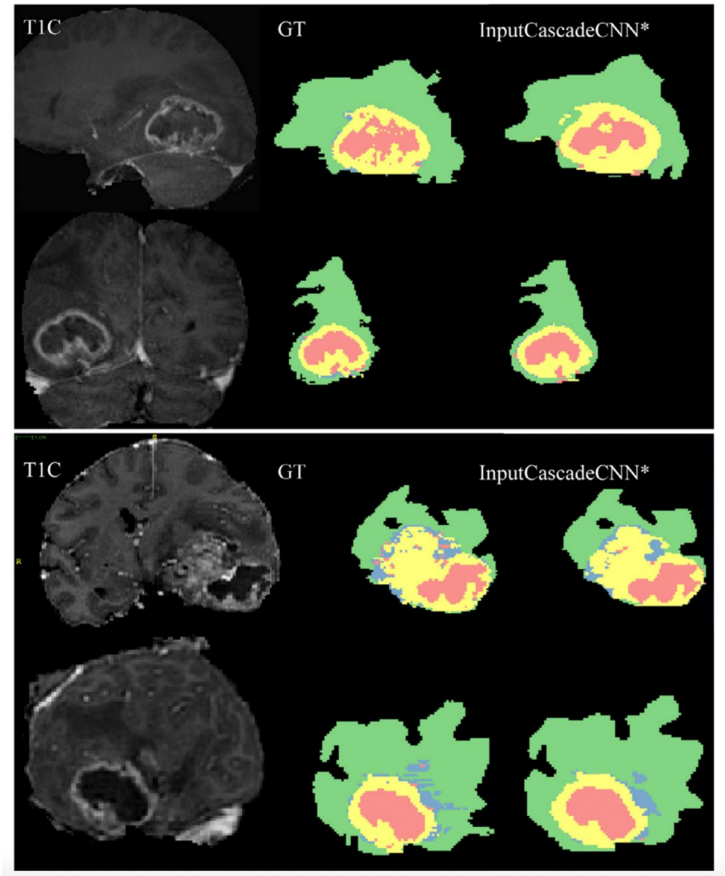

Fig. 5. Visual result from INPUTCASCADECNN on coronal as well as sagittal views. In each sub-figure, the sagittal view is in the top row and the coronal view is in the bottom row. Following are the color codes : edema, necrosis, enhanced tumor, non-enhanced tumor

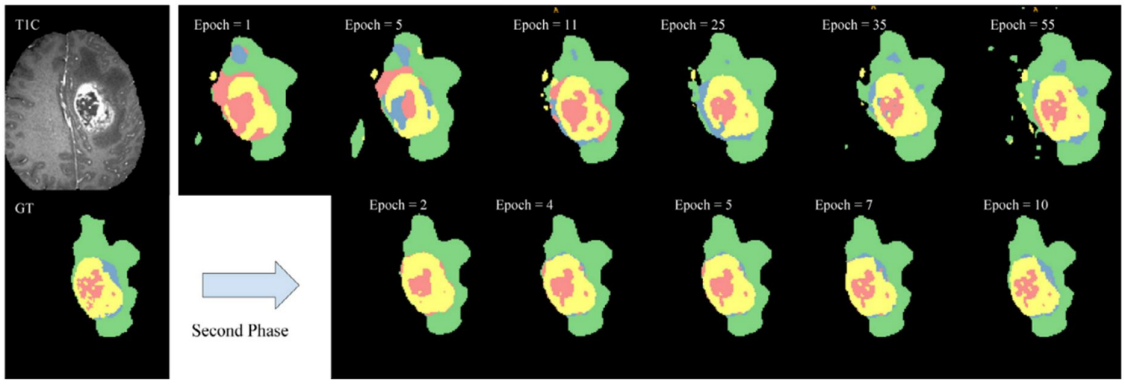

Fig. 6. Movement of learning in INPUTCASCADECNN*. The surge of figures from left to right in the first row depicts the learning process during the first phase. Model will be able to distinguish different tumor subclasses as it learns new features. This is made conceivable because of uniform label distribution of patches during the training of the first phase which causes the model to accept that all classes are equiprobable and causes few false positives. edema, necrosis, enhanced tumor, non-enhanced tumor

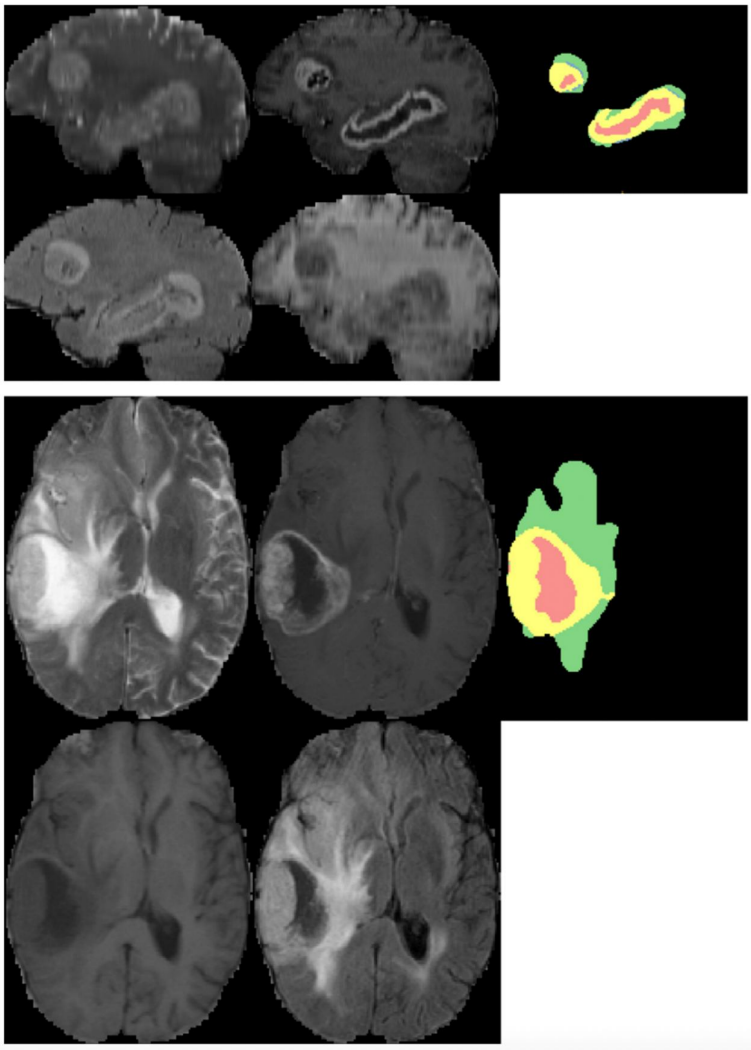

Fig 7. Results of Visual Segmentation from INPUTCASCADECNN, on BRATS 2020 test data-set in axial (bottom) and saggital (top) view. Following are the color codes : edema, necrosis, enhanced tumor, non-enhanced tumor

VI. CONCLUSION

In this paper, deep convolutional neural networks based method has been presented for an automatic brain tumor segmentation. Different architectures have been considered and have investigated the impact of them on the performance. With novel two pathway architecture and by stacking two CNNs for modeling local label dependencies the high performance is achieved. Training is based on a two stage procedure, which allows us to efficiently train CNNs when the label's distribution is unbalanced.

The resulting system of segmentation is very fast due to the use of GPU implementation and convolutional nature of the models. Methods are practical because the time required for segmenting an entire brain with any CNN architecture mentioned varies between 25 seconds to 3 minutes.

REFERENCES

[1] Goodfellow, I.J,iDavid Warde-Farley,iPascaliLamblin,iVincentiDumoulin,iMehdiiMirza,iRazvaniPascanu, iJames Bergstra,iFrédériciBastien,iYoshuaiBengi.iPylearn2:iaimachineilearningiresearchilibrary.iarXivipreprint iarXiv1308.4214

[2] Krizhevsky,iA. ,iSutskever,iI.i,iHinton,iG. i,i2012.iImageNeticlassificationiwithideepiconvolutionalineural inetworks.iNIPS.

[3] Srivastava ,iN.,iHinton,iG.,iKrizhevsky,iA.,iSutskever,iI.,iSalakhutdinov,iR.,i2014.iDropout:ia isimple iway ito ipreventineuralinetworksifromioverfitting.iJ.iMach.iLearn. iiRes. ii15, ii1929–1958.

[4] Goodfellow,iI.J,iWarde-Farley,iLamblin,iP,iDumoulin,iV,iMirza,iPascanu,iBergstra,iJ,iBastien,iBengio,i2013. iPylearn2 i: iA machineilearningiresearchilibrary. iarXivipreprintiarXiv i: i1308.4214.

[5] Menze,iReyes,iM,iB,iLeemput,iK.V,i2021.iTheimultimodalibrainitumoriimageisegmentationibenchmark i(brats). iIEEE iTrans. iMed. iImag.. ii

[6] Farahani,iK.,iMenze, iB.,iReyes, iM., ii2014. iiBrats ii2014 iiChallenge iiManuscripts.

[7] Zeiler,iM.D,iFergus,iR,i2014.iVisualizingiandiunderstandingiconvolutionalinetworks.iIni:iComputeriVisioni-iiECCVi2014.iSpringer,ipp.i818-i833.

[8] Farahani,iK,iMenze,iB,iReyes,iM,i2013.iMultimodalibrainitumorisegmentationi(BRATSi2020).

[9] Meier,iR,iBauer,iS,iSlotboom,iJ,iWiest,iR,iReyes,iM,i2014.iAppearanceiandicontext-sensitiveifeaturesiforibrainitumorisegmentation.iIniProciofiBRATSiChallengei-iMICCAI. ii

[10] Rumelhart,iD.E.,iHinton,iG.E.,iWilliams, R.J.,i1988.iLearningirepresentationsibyibacki-ipropagatingierrors. iiCognit.iMode.i5. ii

[11] Subbanna,iN.,iPrecup,iD.,iArbel,iT.,i2020.iIterativeimultilevelimrfileveragingicontextiandivoxeliinformationiforibrainit umorisegmentationiinimri. ii

[12] Urban,iG.,iBendszus,iM.,iHamprecht,iF.,iKleesiek,iJ.,i2020.iMulti-modalibrainitumorisegmentationiusingideepiconvolutionalineuralinetworks.iProc.iBRATS-MICCAI. ii

[13] Zikic,iD.,iIoannou,iY.,iBrown,iM.,iCriminisi,iA.,i2014.iSegmentationiofibrainitumoritissuesiwithiconvolutionalineurali networks.iProc.iBRATSMICCAI.

[14] Bengio.iY.,iCourville,iA.,iVincent,iP.,i2013.iRepresentationiLearningi:iAireviewiandinewiperspectives.iPatterniAnal.i Mach.iIntell.iIEEEiTrans.i35,i1798-1828.

[15] LeCun,iY.,iBottou,iL.,iBengio,iY.,iHaffner,iP.,i1998.iGradient-basedilearningiapplieditoidocumentirecognition.iProc.iIEEEi86, i2278-2324.

[16] Alvarez,iJ.M.,iGevers,iT.,iLeCun,iY.,iLopez,iA.M.,i2012.iRoadisceneisegmentationifromiaisingleiimage.iIn:iProceedin gsiofithei12thiEuropeaniConferenceioniComputeriVisioni-iVolumeiPartiVII.iSpringer-Verlag,iBerlin,iHeidelberg,ipp. 376–389. i

[17] Long,iJ.,iShelhamer,iE.,iDarrel,iT.,i2015.iFullyiconvolutionalinetworksiforisemanticisegmentation.iCVPR .

[18] Hariharan,iB.,iArbelaez,iP.,iGirshick,iR.,iMalik,iJ.,i2014.iSimultaneousidetectioniandisegmentation.iIn:iComputeriVisi on-ECCVi2014.iSpringer,ipp.i297-312.

[19] Ciresan,iD.,iGiusti,iA.,iGambardella,iL.M.,iSchmidhuber,iJ.,i2012.iDeepineuralinetworksisegmentineuronalimembrane siinielectronimicroscopyiimages.iIn:iAdvancesiiniNeuraliInformationiProcessingiSystems,ipp.i2843-2851.

[20] Goodfellow,iI.J.,iWarde-Farley,iD.,iMirza,iM.,iCourville,iA.,iBengio,iY.,i2013.iMax-outinetworks.iICML. ii

[21] Bauer,iS.,iNolte,L.-P.,iReyes,iM.,i2011.iFullyiautomaticisegmentationiofibrainitumoriimagesiusingisupportivectorimachineiclassificationii nicombinationiwithihierarchicaliconditionalirandomifieldiregularization..iIn:MICCAI,Vol.i6893,ipp.i354-361.

[22] Bauer,iS.,iWiest,iR.,iNolte,iL.,iReyes,iM.,i2013.iAisurveyiofimri-basedimedicaliimageianalysisiforibrainitumoristudies.iPhys.iMed.iBiol.i58,i97-129. i

[23] Angelini, iE., iClatz, iO.E., iKonukoglu, iE., iCapelle, iL., iDuffau, iH.,i2007. iGlioma idynamics iand icomputationalimodels:iAireviewiofisegmentation,iregistration,iandiinisilicoigrowthialgorithmsianditheiriclinicaliapplic ations. iCurr.iMed.iImagingiRev.i3 i(4), i262–276.

[24] Clark, iM., iHall, iL., iGoldgof, iD., iVelthuizen, iR.P., iMurtagh, iF.,iSilbiger, iM.L., i1998. iAutomatic itumor isegmentation iusing iknowledgebased iclustering. iIEEE iTrans. iMed. iImag. i17, i187–201.

[25] Doyle,iS.,iVasseur,iF.,iDojat,iM.,iForbes,iF.,i2013.iFullyiautomaticibrainitumorisegmentationifromimultipleimrisequences iusing ihidden imarkov ifields iand ivariational iem. iProc. iBRATSMICCAI.
[26] Prastawa, iM., iBullit, iE.,iHo,iS., iGerig, iG., i2004. iA ibrain itumor isegmentation iframework ibased ion ioutlier idetection. iMed. iImage iAnal.i8,i275–283.
[27] Prastawa,iM.,iBullitt,iE.,iHo,iS.,iGerig,iG.,i2003.iRobustiestimationiforibrainitumorisegmentation.iIn: iMedical iImageiComputingiand iComputerAssisted iIntervention-MICCAI i2003. iSpringer, ipp. i530–537. ii
[28] Khotanlou, iH., iColliot, iO., iAtif, iJ., iBloch, iI., i2009. i3D ibrain itumor isegmentationiin imri iusing ifuzzy iclassification, isymmetry ianalysis iand ispatially iconstrained ideformable imodels. iFuzzy iSets iSyst. i160, i1457–1473.
[29] Cobzas, iD., iBirkbeck, iN., iSchmidt, iM., iJgersand, iM., iMurtha, iA., i2007. i3D ivariational ibrain itumor isegmentation iusing ia ihigh idimensional ifeature iset. iIn: iICCV, ipp. i1–8.
[30] Popuri, iK., iCobzas,., iMurtha, iA., iJgersand, iM., i2012. i3D ivariational ibrain itumor isegmentation iusing idirichlet ipriors ion ia iclustered ifeature iset.. iInt. iJ. iComput. iAssist. iRadiol. iSurg. i7, i493–506.i
[31] Kwon, iD., iAkbari, iH., iDa, iX., iGaonkar, iB.,, iC., i2014. iMulti-modal ibrain itumor iimage isegmentation iusing iglistr. iIn: iin iProceedings iof iBRATS iChallenge i- iMICCAI. ii
[32] Parisot, iS., iDuffau, iH., iChemouny, iS., iParagios, iN., i2012. iJoint itumor isegmentation iand idense ideformable iregistration iof ibrain imr iimages.. iIn: iMICCAI, iVol. i7511, ipp. i651–658.
[33] Havaei, iM., iJodoin, iP.-M., iLarochelle, iH., i2014. iEfficient iinteractive ibrain itumor isegmentation ias iwithin-brain iknn iclassification. iIn: iInternational iConference ion iPattern iRecognition i(ICPR).
[34] Hamamci, iA., iKucuk, iN., iKaraman, iK., iEngin, iK., iUnal, iG., i2012. iTumor-cut: iSegmentation iof iibrain itumors ion icontrast ienhanced imr iimages ifor iradiosurgery iapplications. iIEEE iTrans. iMed. iImag. i31, i790–804.
[35] Kleesiek, iJ., iBiller, iA., iUrban, iG., iKothe, iU., iBendszus, iM., iHamprecht, iF.A., i2021. iilastik ifor imulti-modal ibrain itumor isegmentation. iProc. iBRATS-MICCAI. ii
[36] Meier, iR., iBauer, iS., iSlotboom, iJ., iWiest, iR., iReyes, iM., i2014. iAppearance- iand icontextsensitive ifeatures ifor ibrain itumor isegmentation. iin iproc iof iBRATS iChallenge i- iMICCAI.
[37] Subbanna, iN., iPrecup, iD., iCollins, iL., iArbel, iT., i2013. iHierarchical iprobabilistic igabor iand imrf isegmentation iof ibrain itumors iin imri ivolumes.. iIn: iin iProceedings iof iMICCAI, iVol. i8149, ipp. i751– 758. ii
[38] Tustison, iN., iWintermark, iM.C.D., iAvants, iB., i2013. iAnts iand iárboles. iIn: iin iProceedings iof iBRATS iChallenge i- iMICCAI.
[39] Doyle, Schmidt, iM., iLevner, iI., iGreiner, iR., iMurtha, iA., iBistritz, iA., i2005. iSegmenting ibrain itumors iusing ialignment-based ifeatures. iIn: iInt. iConf ion iMachine iLearning iand iApplications, ipp. i6–pp.
[40] Lee, iC.-H., iSchmidt, iM., iMurtha, iA., iBistritz, iA., iS, iJ., iGreiner, iR., i2005. iSegmenting ibrain itumor iwith iconditional irandom ifields iand isupport ivector imachines. iIn: iin iProceedings iof iWorkshop ion iComputer iVision ifor iBiomedical iImage iApplications.
[41] Zikic, iD., iGlocker, iB., iKonukoglu, iE., iCriminisi, iA., iDemiralp, iC., iShotton, iJ., iThomas, iO., iDas, iT., iJena, iR., iPrice, iS., i2012. iDecision iforests ifor itissue-specific isegmentation iof ihigh-grade igliomas iin imulti-channel imr. iIn: iMedical iImage iComputing iand iComputer-Assisted iIntervention– iMICCAI i2012. iSpringer, ipp. i369–376.
[42] Gotz, iM., iWeber, iC., iBlocher, iJ., iStieltjes, iB., iMeinzer, iH.-P., iMaier-Hein, iK., i2021. iExtremely irandomized itrees ibased ibrain itumor isegmentation. iIn: iin iProceedings iof iBRATS iChallenge - iMICCAI.
[43] Davy, iA., iHavaei, iM., iWarde-Farley, iD., iBiard, iA., iTran, iL., iJodoin, iP.-M., iCourville, iA., iLarochelle, iH., iPal, iC., iBengio,iY., i2014. iBrain itumor isegmentation iwith ideep ineural inetworks. iProc. iBRATS-MICCAI.
[44] Pinheiro, iP., iCollobert, iR., i2014. iRecurrent iconvolutional ineural inetworks ifor iscene ilabeling. iIn: iProceedings iof ithe i31st iInternational iConference ion iMachine iLearning, ipp. i82–90. i
[45] Farabet, iC., iCouprie, iC., iNajman, iL., iLeCun, iY., i2013. iLearning ihierarchical ifeatures ifor iscene labeling. iPattern iAnal. iMach. iIntell. iIEEE iTrans. i35, i1915–1929.
[46] Huang, iG.B., iJain, iV., i2013. iDeep iand iwide imultiscale irecursive inetworks ifor irobust iimage ilabeling. iarXiv ipreprint iarXiv:1310.0354.