3D Brain Tumor Segmentation: Narrow UNet CNN

Mathew Camp[1], Kenneth Gonzalez[2], and Miaomiao Zhang[3]
Georgia Gwinnet College[1], University of Puerto Rico at Arecibo[2], and Lehigh University[3]
Email: {miaomiao, mac618, ksg218}@cse.lehigh.edu

***Abstract*—Reliable methods for tumor detection and brain abnormalities are crucial to help find diseases at early stages. Having accurate software that uses machine learning to identify abnormalities of the brain may prevent a disease progression if used on an MRI of the patient. In this paper, we develop a neural network to detect and highlight brain tumors present in MRI's. Our model is designed to be more compact than typical CNNs to minimize prediction times while still maintaining prediction accuracy. The model uses a dice coefficient for the loss function as well as accuracy metric. We adopt Adadelta as the optimizer as it is more robust and eliminates the requirement of manually tuned learning rates. Our model reduces the prediction time with fewer layers and convolution filters, while allowing rapid convergence to a stable solution. In addition, the models hyper-parameters are being fine-tuned in an iterative process to ideally achieves better segmentation accuracy. Experimental results show that our model improves the performance compared with the state-of-the-art methods.**

***Keywords*-image segmentation, brain tumor detection, neural network**

I. Introduction

Machine learning designs and builds algorithms that are guided by data. Rather than depending on humans to explicitly define instructions, machine learning algorithms use training sets of real-world data to generate new data, known as predictions, that prove to be more accurate than those from models designed by humans. Within the field of machine learning, neural networks are a subset of algorithms built around a model of artificial neurons spread across many interconnected layers. Within neural networks, deep learning describes complex and deeper networks than usual. The advantage of these added layers is that the networks are able to develop much great levels of abstraction. This is necessary for specific tasks, like image recognition and automatic translation which are very complex [8].

A Convolutional Neural Network is part of the category of Neural Networks that are used with image data. These have been successful in identifying faces, objects, traffic signs, as well as powering machine vision in robots and self-driving cars [15]. CNN's have been applied successfully on a variety of biomedical segmentation problems. Most existent approaches relied on using 2D CNN's for processing 3D volumes, due to difficulties being reported when training with whole 3D volumes. There 2D architectures might be successful in some cases but they are suboptimal in their use of already existent 3D information.

When we began, a base code from an existent 3D UNET was used for testing. It was suggested to use a deep network that learns to generate dense volumetric segmentations, but only requires annotated 2D slices for training. This network could be used in two different scenarios, the first application just aims on densification of a sparsely annotated data set; the second one learns from multiple sparsely annotated data sets to generalize to new data [13]. Like us, there are different researchers working on the segmentation of images for different applications. Brain disease research is among the highest searched for studies to find better algorithms for the segmentation or prediction of these diseases. CNNs are used to segment specific diseases for example cancer, lesion, and alzheimer disease; building deep neural networks to scan specific areas of the brain in both patch-wise and scanning through a whole 3D volume are different ways these researchers handled the predictions of these diseases with a CNN [3, 9, 10, 11]. Among other studies to make disease segmentation easier is the brain segmentation, extraction or skull stripping, which cleans the MRI and leaves only the brain [7, 14]. CNNs are also used for everyday use research like image classification and modelling sentences, which actually makes classification tasks easier with a computer [4, 12].

With our CNN, we use more than one of these useful applications some researches have made such as skull stripping [7, 14]. This reduces complexity for the analysis of the training data. If the skull was not removed we could get misidentified tumors, slower performance, longer training times, and worse results. We validate the proposed model on two data sets: the first is the BRaTS data set [6, 5] and clinical data collected from neurosurgery. Both sets contain T1 weighted MRIs. The BRaTS data set of course was made experts, while we made the best to make our manually masked set be similarly masked to the BRaTs, on a different set of data. The use of more than one set and data augmentation reduces the probability of overfitting in our model. The 3D volumes are split into 2D slices and the resulting images are resampled, standardized, and normalized so that the model is fed consistent data input. The patient samples are chosen at random at the start of the training in an 80/20 split for training and validation.

II. Architecture

We developed a customized, highly iterative UNet that allow for rapid prototyping and testing of designs. The models are generated dynamically based on architecture settings such as depth of the network, segmentation levels for encoders and decoders, activation functions for each block, kernel initializers and constraints. All settings are easily modified, and the changes are reflected in the model summary Keras generates when run [2].

Many existing medical imaging segmentation neural networks, including the leading 3D CNN UNet, rely on networks dozens of layers deep with many millions of trainable parameters. These models are very flexible in their ability to be used on a variety of different images and segment different features while being able to produce results with impressive accuracy. The goal of our paper is two-fold: develop a neural network that could produce 3D volumes of tumors from brain MRIs, but also to find an alternative design to these vast, extremely deep networks. This goal is inspired by just how long some of these models took to generate a prediction, let along train. Medical image segmentation should be performed as quickly as possible without sacrificing accuracy. Training times are not as important as prediction times, but training should also be a relatively quick process in case the network should need to be retrained with new data.

A shallow neural network has clear disadvantages over deeper networks of similar design due to an objectively fewer number of training parameters. To overcome this obstacle and create a network that can produce very quick and accurate results, we take full advantage of every convolution filter. As seen in figure 1 we

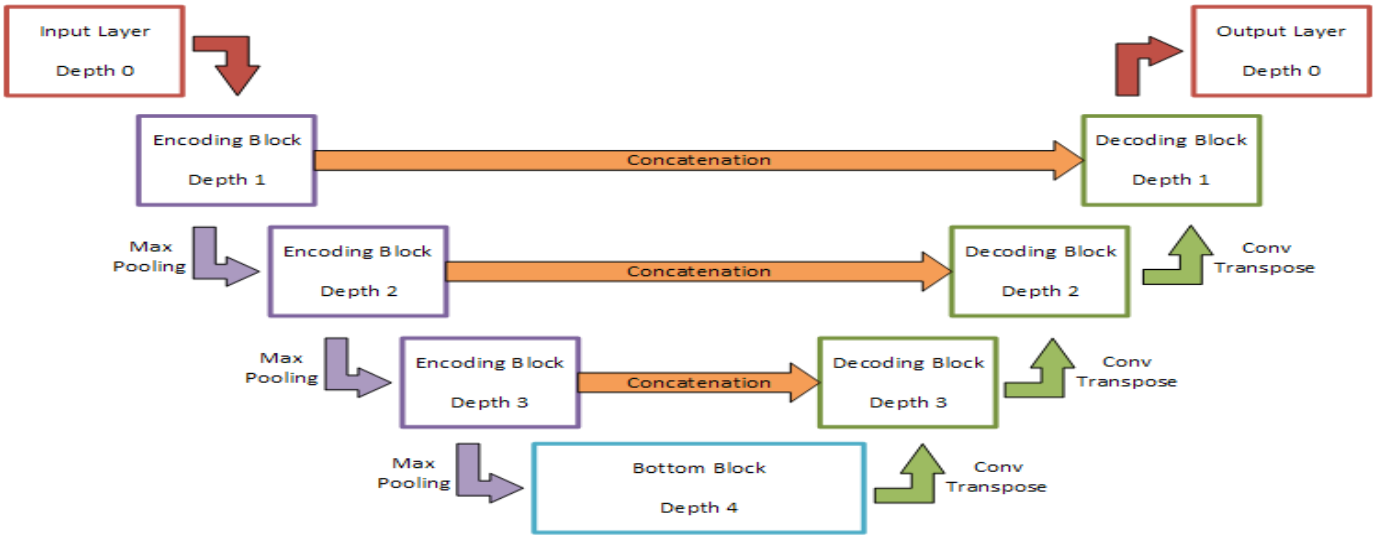

Fig. 1. The Architecture of our Model

strategically designed the encoding and decoding blocks of our UNet model with the idea of eliminating co-adaptation.

Inspired by [13] our work employed 3D varieties of convolution, max pooling, and deconvolution/upsampling layers. We believe that this increase in dimensionality, while clearly effective, was unnecessary if the network is crafted just so. Our model uses conventional 2D layers and runs through axial slices of an MRI to create a 3D volume instead. This reduction in dimensionality means that instead of training on 128^3 voxels per sample, the network trains on 128^2 pixels per slice. This results in reduced preprocessing, training, and prediction times.

III. Optimization

A. Loss function

In order for the optimizer to properly evaluate and back-propagate error, we use the dice coefficient (similar to an intersection over union) as a metric to measure the accuracy of the models predictions. A typical accuracy metric quickly proved useless for this image segmentation task as the vast majority of the image was marked as background (zero on the ground truth data). A binary accuracy metric counts the number of pixels, or in this case voxels, are correct, including background data. Rewarding the model for correctly tagging the background as well as nontumor areas as not being a tumor is proved to lead to very suboptimal solutions.

The problem of too much rewarding was solved by using a slightly modified dice coefficient as the loss function as shown in equation below. The loss function does not reward the network for correctly predicting background, it only is rewarded for correctly predicting features. The network is then received negative reinforcement for false positives and false negatives. This creates a much more conservative measurement for accuracy of the network. For example, during one training step the network reported an accuracy of 0.9847, but the dice coefficient was only 0.6931. Having a metric that punishes for false positives instead of rewarding for true negatives allows for the optimizer to follow a much more reliable gradient.

$$\begin{aligned} \mathrm{T} &= vec(\mathrm{Truth}) && \text{vectorize ground truth} \\ \mathrm{P} &= vec(\mathrm{Pred}) && \text{vectorize prediction} \\ \mathrm{DSC} &= \frac{2 \cdot \sum (T \cdot P) + \epsilon}{\sum \mathrm{Truth} + \sum \mathrm{Pred} + \epsilon} && \epsilon\text{=0.00001} \\ \mathrm{Loss} &= 1 - \mathrm{DSC} \end{aligned}$$

The loss function is computed by first taking the ground truth and prediction tensors and vectorizing or flattening them. This operation keeps each value paired up properly between the truth and prediction tensors. Next, the dice coefficient is computed with a smoothing factor as shown in the equation. Finally, to transform the dice coefficient (ranging from 0 to 1) to a loss function (ranging from 1 to 0), the dice coefficient is subtracted from 1. This allows smaller values to reflect better predictions, ideal for how a loss function should behave.

B. Optimizer

Another attempt made to overcome the inherent drawbacks of a shallow network was to use the ADADELTA optimizer, an alternative to the standard preferred ADAM optimizer [1]. ADADELTA was chosen because of its low computational costs over SGD and the dynamic adjustments it makes. In addition, ADADELTA does not have the learning rate decay issue that ADAGRAD faces. ADADELTA dynamically regulates its learning rate, and thus does not require extensive guesswork of manual learning rate scheduling. ADADELTA begins with an extremely aggressive learning rate default of 1.0 as determined from the Keras source code. This allows the network to quickly follow the gradient before the learning rate is adjusted by the optimizer and the network fine tunes over time. Our model, specifically, required only a few epochs of training on the BRaTS dataset before it reached a stable solution that then required fine-tuning over the following epochs.

IV. Experiments

A. Data

The neural network was trained on the BRaTS dataset with preoperative TCGA BGM and TCGA LGG images and segmentations. Due to time restraints in this project, only the post-contrast T1 weighted MRI was used for training. The neural network may very

well perform better with all modality data instead of just the post-contrast images. Testing was performed with T1 weighted MRIs from the Brigham and Womens Hospital clinical data of 15 patients with brain tumors.

B. Preprocessing

Information contained in the MRIs contained in the BRaTS dataset used and the clinical dataset provided by Brigham and Womens Hospital are not consistent with each other in terms of data ranges and even data types stored in the MRI files. A neural network performs best when input data ranges are relatively similar, thus causing neurons to activate in similar manners from one input sample to the next.

To give the network the best chance at high performance, the MRIs are all resampled to a cube with sides of 128 voxels in length. The MRI data is then sliced along the axial view and portioned into batches of 60 for the network to process. Each batch of 75 MRI axial slices is normalized to have a mean of 0 and standardized so that voxel values are standard deviations instead of their raw data. The mean and standard deviation are obtained feature-wise through the Keras ImageDataGenerator [2]. Since all of the BRaTS data is within similar ranges of values, this method of normalizing and standardizing allows the network to accept any self-consistent MRI data to be used as input regardless of the absolute range of values contained in the MRI. By removing the skull from the MRI scan, leaving a clean brain, there is reduced complexity for analysis of the training data. Failing to remove the skull and preprocess input data results in worse performance, longer training times, and misidentification of tumors in the brain.

Despite the BRaTS dataset having relatively consistent MRI data, there is an issue of contrast varying from one patient to the next. The normalization and standardization help alleviate brightness differences, but they only exacerbate the difference in contrast. To solve this, each slice is processed through an adaptive histogram equalization algorithm that boosts contrast and highlights subtle features in the image. We aim to maintain uniform contrast between MRIs and expose details that the neural network may choose to pick up on.

We use Keras for image augmentation to artificially generate more data from the MRI slices used for training and validating. There are configuration options for applying random rotations, horizontal and vertical shifts, shearing, and other augmentations. The slices are augmented in real-time on the CPU and then used in training on a single or multiple GPUs. This augmentation process greatly increases the diversity of images with which the network can use to train. Care was taken not to distort the images too greatly such that the brains become unrecognizable, as such augmentation could negatively impact training.

C. Training

The UNet is trained in batches of 75 images that are 128^2 in size containing only a single grayscale channel. The images are chosen in a shuffled order for each epoch and augmented randomly from one epoch to the next by the preprocessing. The BRaTS dataset was used as input to the network after an 80/20 split for training and validation purposes. This split was chosen on a slice by slice basis as opposed to splitting patients by this ratio. This method of splitting gives the network portions of MRIs that can be distributed between both training and validation subsets.

During the development phase of the project, the neural network was trained with a version of TensorFlow that supported SSE4.1, SSE4.2, FMA, and AVX instructions. Iterative training was performed on an Intel i7-7700HQ CPU using either 6 of the 8 logical cores, or all 8 logical cores when the computer was not in use. Fully-fledged training was performed 2 nodes of a cluster, each with 2 Nvidia GTX 1080 cards and 2 12-core Xeon E5-2670 CPUs. The cluster nodes were only used for training once we saw the model training follow a favorable direction on a personal computer.

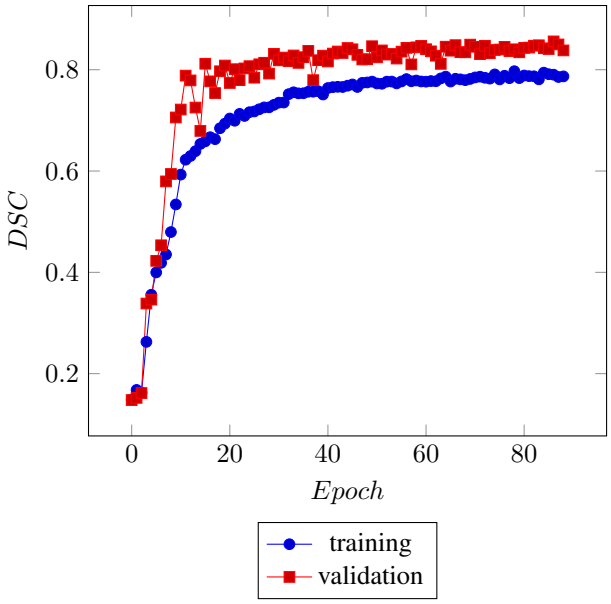

Fig. 2. Training and Validation Graph

Due to the black box nature of neural networks, custom mini prediction code was written to provide rapid prediction previews both after each training step and epoch. A mini prediction is one or more slices of an MRI that are chosen manually as being representative of the one of the test brains. Predictions were performed on these slices only and shown to the user. These previews aided in visualizing what the network was doing and gave a sense of directionality that allowed further iterative changes to be made to the model. Metrics were also saved to a file after each training step to provide insight in to network performance quickly without having to wait for several epochs to discover a trend. Rough estimations of a potential trend could be made based on training step metrics and they were used to quickly change the model if unexpected behavior occurred.

D. Evaluation

Quick predictions, as mentioned, are generated optionally after each training step as well as after each epoch. These predictions were used as the first level of testing for our network model. Instead of splitting the valuable BRaTS data between training, validation, and testing, we opted to use an entirely different dataset for testing. In theory, the robustness of our model would be determined by how well it handles a different dataset.

A downside of testing with our clinical data is the lack of an expert provided ground truth for brain tumor segmentation. Our results data metrics are therefore limited to using validation data from the BRaTS dataset.

V. Results

Two distinct datasets were used for training: our data set and the BRaTS dataset. Both datasets contain T1 weighted MRIs. The BRaTS dataset contains image masks created by experts, while we manually masked our set. The use of two datasets as well as image augmentation reduces the chances of over-fitting the model. The 3D volumes are split into 2D slices along the axial orientation and the resulting images are resampled, standardized, and normalized so that the model is fed consistent input data. Patient MRIs are randomly

chosen at an 80/20 split for training and validation and only image slices that contain tumors are used by the network.

Due to the lack of ground truth data in the clinical dataset, the results shown rely on validation data from the BRaTS dataset. Twenty percent of the axial slices were used from validation only and the scores were calculated using the dice coefficient formula show before. After a short training session of only 18 epochs, and then another session of 6 epochs, the network scored 0.8029 on the validation subset (with a raw binary accuracy of 0.9915), details are show in table [1]. Upon further inspection, it would appear most of the error in the predictions come from false negatives near the bottom of the brain. We believe this is because the network has virtually no knowledge of cerebellum, which has a different texture and gradations of gray in it that the cerebral cortex. Most of the tumors marked in the BRaTS dataset are within the cerebral cortex, causing a lack of knowledge and throwing of the network with completely new data that the convolutional layers are likely unsure of what to do with.

This training session only spanned a few hours, but the ADADELTA optimizer converged quickly on a solution in only 8-10 epochs. The validation score remained greater than the training score for nearly the entire time, which leads us to believe there is some improvement to be made to the model or preprocessing.

Shown in figure 3 are four examples of predictions outputted by the neural network. The red outline shows the region the network selected as containing a tumor. Despite these images being just two-dimensional slices, the network does save the volumetric segmentation alongside the original input MRI data, both resampled to the same cubic size. Most of the time, the networks segmentation is spot on. There are only a few cases of false negatives by the network, but they occur on the very edge slices (top and bottom) of the tumor in some situations. The network also struggles to properly segment tumors that are both dark and near the edge of the brain.

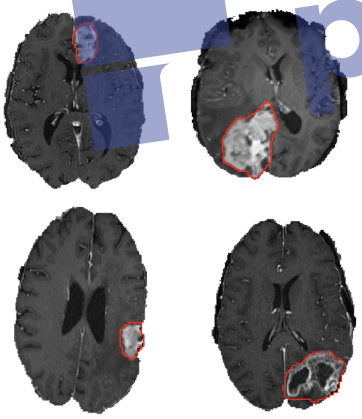

Fig. 3. Sample Predictions

VI. Acknowledgments

This material is based upon work supported by the National Science Foundation under grant no. CNS-1757787.

Validation Subset Dice

Method	Whole Tumor
DeepMedic	0.89
Kamnitsas et. al.	0.85
Two Path CNN	0.85
Narrow UNet	**0.83**
Pereira et. al.	0.78

Table 1: Comparison with other successful neural networks

References

[1] Adam optimizer. URL https://github.com/tensorflow/tensorflow/blob/master/tensorflow/python/training/adam.py#L39.

[2] Keras documentation. URL https://keras.io/preprocessing/image/#imagedatagenerator-class.

[3] Giovanni Montana Adrien Payan. Predicting alzheimer's disease: a neuroimaging study with 3d convolutional neural networks, 2015. URL https://arxiv.org/abs/1502.02506.

[4] Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. Imagenet classification with dcnn, 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[5] Carole Sudre Lucas Fidon Dzhoshkun I. Shakir Guotai Wang Zach Eaton-Rosen-Robert Gray Tom Doel Yipeng Hu Tom Whyntie Parashkev Nachev Marc Modat Dean C. Barratt Sbastien Ourselin M. Jorge Cardoso Tom Vercauteren. Eli Gibson*, Wenqi Li*. "niftynet: a deep-learning platform for medical imaging." computer methods and programs in biomedicine, 158 (2018): 113-122., 2018. URL https://arxiv.org/pdf/1709.03485.

[6] Sebastien Ourselin Tom Vercauteren. Guotai Wang, Wenqi Li. "automatic brain tumor segmentation using cascaded anisotropic cnn." in brainlesion: Glioma, multiple sclerosis, stroke and traumatic brain injuries. pages 179-190., 2018. URL https://arxiv.org/abs/1709.00382.

[7] Alexander Hubert Daniel Schwarz Klaus Maier-Hein Martin Bendszus Armin Biller Jens Kleesiek, Gregor Urban. Deep mri brain extraction: A 3d convolutional neural network for skull stripping, 2016. URL https://www.sciencedirect.com/science/article/pii/S1053811916000306.

[8] Tyler Keenan. Neural networks: Machine learning inspired by the brain, 2018. URL www.upwork.com/hiring/data/neural-networks-demystified/.

[9] Christian Ledig Daniel Rueckert Konstantinos Kamnitsas, Liang Chen and Ben Glocker. Multi-scale 3d cnn for lesion segmentation in brain mri, 2015. URL hhttp://www.isles-challenge.org/ISLES2015/pdf/20150930_ISLES2015_Proceedings.pdf#page=21.

[10] Chris Pal Hugo Larochelle Pierre-Marc Jodoin Mohammad Havaei, Francis Dutil. A convolutional neural network approach to brain tumor segmentation, 2015. URL https://link.springer.com/chapter/10.1007/978-3-319-30858-6_17.

[11] David Warde-Farley Antoine Biard-Aaron Courville Yoshua Bengio Chris Pal-Pierre-Marc Jodoin Hugo Larochelle Mohammad Havaei, Axel Davy. Brain tumor segmentation with deep neural networks, 2016. URL https://www.sciencedirect.com/science/article/pii/S1361841516300330.

[12] Adrinne M. Mendrik Nal Kalchbrenner, Edward Grefenstette. A convolutional neural network for modelling sentences, 2014. URL https://arxiv.org/pdf/1404.2188.pdf.

[13] Soeren S. Lienkamp Thomas Brox Ozgun Cicek, Ahmed Abdulkadir and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016. URL https://lmb.informatik.uni-freiburg.de/Publications/2016/CABR16/cicek16miccai.pdf.

[14] Phil Blunsom Linda S. de Vries Manon J. N. L. Benders-Ivana Igum Pim Moeskops, Max A. Viergever. Automatic segmentation of mr brain images with a cnn, 2016. URL https://ieeexplore.ieee.org/document/7444155/.

[15] Ujjwalkarn. An intuitive explanation of convolutional neural networks, 2017. URL ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets.