MRI Glicoma Tumor Segmentation with U-NET

Sasha Yousefi
Stanford University
450 Serra Mall, Stanford, CA 94305
syousefi@stanford.edu

Jessica Houghton
Stanford University
450 Serra Mall, Stanford, CA 94305
jessicahoughton@stanford.edu

1. Abstract

Manual tumor segmentation is labor intensive and oftentimes results in disagreements and mistakes among clinicians. Ideally, if a model were able to more accurately segment tumor tissue from MRI images there could be an increase in patient health and longevity in those diagnosed with glioma. The current standard for medical image segmentation is the U-NET, which is able to extract semantic information while retaining structural integrity of the image. In this project, we present an improved U-NET architecture for glioma segmentation using images from the MICCAI BraTS 2020 dataset. We expand upon the existing U-NET adding VGG16 and VGG19 structures to the existing model layout. Through this addition, data augmentation and regularization techniques, we achieve superior model performance which outperforms the Support Vector Machine and basic U-NET baseline methods. A dice score of 0.65 on the test set indicates that our model achieves promising segmentation results. Future research into elastic deformations, spatially adaptive normalization techniques, and transfer learning are necessary.

2. Introduction

We will be investigating automated tumor segmentation for brain tumors based on MRI images from the BRATS 2020 dataset [2]. The current process of manual segmentation is labor-intensive and there are often disagreements and/or mistakes. Thus, there is a need for an automatic and robust segmentation system. We will be using the BraTS dataset which is publicly available on Kaggle [1]. This dataset contains about 368 MRI participants with tumors (368 glioma cases = 1,472 mpMRI scans) and manually labeled segmentations. Each segmentation map contains 4 classes (0: no tumor, 1: edema, 2: necrotic core/non-enhancing, 3: enhancing) which are distinct tumor cells with different gene expression profiles that make up a glioma. Given MRI scan images, the models described in this paper will perform a 4-class classification for each pixel within the image in order to create a prediction mask.

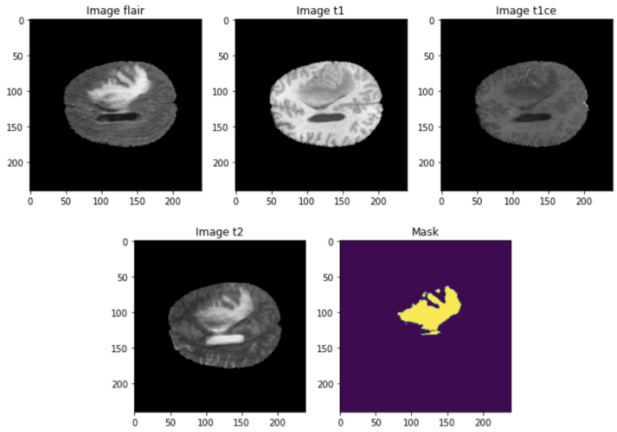

Figure 1. Brain Image MRI scans. Images 1-4 show the brain/tumor complex from the four different types of MRI scans. Image 5 shows the example map that was manually identified by technicians.

In our project, we proposed the use of U-NET, which builds upon the Fully Connected Convolutional Neural network in order to yield precise segmentations for datasets with limited data [15]. The current issue with Fully Connected CNNs for our dataset is the following. First, the network is quite slow since it must be run independently for each patch. Additionally, there is a fair amount of redundancy due to the overlapping patches for each image. Second, the use of max-pooling layers results in a trade-off between localization accuracy and image context. Larger patches which require more pooling layers lose localization accuracy, while smaller patches with less pooling layers lose image context.

In order to remedy these shortcomings, U-NET provides the following solution. The U-NET architecture consists of a contracting path and an expanding path (explicitly described in Methods). The contracting layer learns the feature maps for the image and reduces the image down to a feature vector, and the expanding layer then reconstructs the image from the feature vector. However, this technique

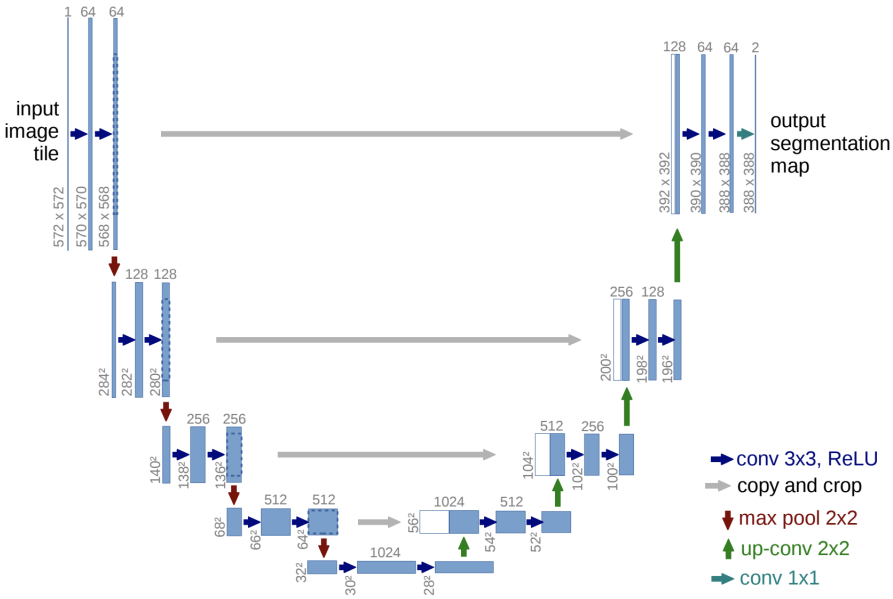

Figure 2. Architecture of U-NET. Step by step description of layers in Methods. [3]

alone loses structural integrity during image reconstruction. In order to maintain structural nuances of the image, the U-NET utilizes the feature mappings learned at each step of the contracting path to expand the vector into a segmented image [3]. Thus, the maps learned in the each contracting path are concatenated with the expanding path (coined skip connections), allowing the expanding path to make use of prior structural information. Additionally, the creators of U-NET also added a large number of feature channels, which allow the network to propagate more information into these high resolution layers. U-NET has been highly successful in segmentation tasks for biomedical images, and is our primary architecture for predicting brain tumor segmentations.

3. Related Work

Previous work investigates using machine learning methods such as support vector machines, k nearest neighbors, and random forests for brain tumor segmentation [9]. These models are simple compared to deep learning models, yet have proven effective in segmentation tasks. SVMs are considered state-of-the-art classifiers that work well in high dimensions and are relatively memory efficient [5]. Typical linear SVM studies achieved promising results of 0.51, 0.35, 0.45 Dice coefficients for complete, core, and enhancing tumor types. These results have been improved with more complicated SVM models such as with a radial basis function kernel or using spatial coordinate features. Our baseline implementation will explore the simple linear SVM classifier, which uses linear boundaries to classify pixels by maximizing the margin between classes.

Work in this area of deep learning in medical imaging also uses Baseline Convolutional Neural networks (BCN) and Fully Convolutional Neural Networks (FCN) for segmentation tasks [16]. The architecture for these BCNs include computing pixel wise probabilities for each subregion of the image. Typical BCNs use the CNN architecture with 3D convolutional kernels. Typically, each convolutional layer is followed by a ReLU nonlinearity. The final few layers are often fully-connected layers whose outputs are 0/1 binary values for whether a pixel in the subregion belongs to the tumor. Shen et al., shows that this baseline model achieves high accuracy (84.5 percent). However, fully connected layers are not the most efficient method for medical segmentation tasks as they require a large amount of parameters to be tuned.

Other related work also explores the use of VGG16 U-NET hybrids with transfer learning and dropout. The choice of VGG16 in this case was optimal since its structure it quite similar to the U-NET's contracting layer. The dropout helps regularize the model, as the nature of VGG16 contains many nonlinear hidden layers and complex relationships that often lead to overfitting [12]. Researchers were able to achieve pixel accuracies of 0.994 and 0.9975 from basic U-Net and improved VGG16 U-Net architectures. Although these accuracies both seem large and the VGG16 U-NET seems like a small improvement over the plain U-NET, this is substantial in our case since the majority of the pixels in the image are non-tumor pixels. Thus, the marginal increase in accuracy reflects the model being able to better segment glioma related pixels [8]. The use of transfer learning was also prevalent in these studies to help speed

up training and reduces computing time. Transfer learning freezes the contracting layer in the VGG16 U-NET so that the weighted layer is not updated when evaluating training data. Instead, we utilize the weight of the convolution layer of the VGG16 model in place of these contraction layer weights [12]. An additional benefit of the transfer learning approach also is that it typically produces better results than the random initialization of the training parameters [13].

4. Methods

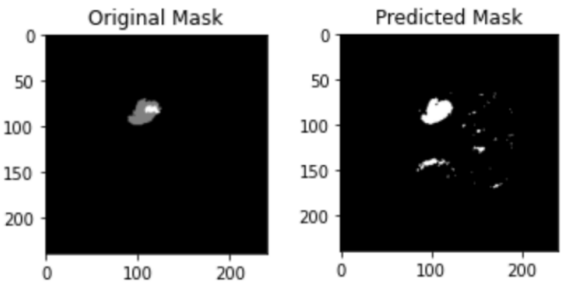

Figure 3. Predicted segmentation task with SVM. The left shows the original tumor. The right shows the predicted mask from our SVM.

In order to compare performance between models, we will use a two metrics commonly used in image segmentation: mean dice coefficient and mean intersection-over-union (IoU). The IoU score (also known as the Jaccard Index) is the area of overlap between the prediction and ground truth over the area of the union between the two. The formula for IoU score is as follows, where P is the set of predictions labeled "1" and T is the set of ground truth pixels labeled "1":

$$IoU\ Score = \frac{|P \cap T|}{|P \cup T|}$$

Since our segmentation model has multi-class predictions we will use the mean-IoU score which simply computes the average of the IoU score for each class.

The dice coefficient is very similar to IoU, except that it divides the overlapping area between the prediction and ground truth labels by the total area. The formula for dice coefficient is as follows, where P is the set of predictions labeled "1" and T is the set of ground truth pixels labeled "1":

$$Dice\ Coefficient = 2\frac{|P \cap T|}{|P| + |T|}$$

Like mean-IoU, we will use the mean dice coefficient which computes the average of the dice coefficient for each class. It is important to note that the IoU and dice coefficient are positively correlated

First, we discuss our baseline method, which is the Support Vector Machine (SVM). In brief, the SVM is a linear classifier that aims to classify data points by finding the hyperplane that maximizes the margin between the decision boundaries of the different classes and their nearest points. Prior to classification, the model extracts first order texture features (mean, variance, skewness, kurtosis, energy, and entropy) from each pixel in each image. Skewness and kurtosis are defined as the third and fourth moment of the standard score, and energy is defined as the following:

$$Energy = \sum_{i=1}^{N}(X(i) + c)^2$$

After extraction of the first order features for each image, the image was centered and each pixel was run through the linear SVM in order to predict whether that pixel belonged to the tumor class or not, creating a predicted mask [18]. For simplicity of our baseline model, we simply ran this method on the "whole tumor" class. In order to quantify model quality, we calculated the dice score between the predicted tumor mask and the actual tumor mask.

The next model we implemented was a basic U-Net for semantic segmentation. A U-Net consists of a contracting and expanding path. The contracting path consists of two sequential 2D convolutional layers with 3x3 filters and "same" padding. "Same" padding is a mechanism to pad the input image such that the output dimensions will have the same spatial dimensions of the input. These dual convolution layers are followed by a ReLU non-linearity and a 2x2 max-pooling operation with a stride of 2. This max-pooling acts as the downsampling operation. At each downsampling step, the number of filters in the 2D convolution layers are doubled and the schema repeats. A dropout of probability d is applied after the last ReLU non-linearity in the downsampling path.

The expanding path consists of an upsampling operation of the feature map, a concatenation operation that concatenates the the upsampled output with the corresponding cropped feature map from the contracting path, two 2D convolutional layers with 3x3 filters and "same" padding, and a ReLU non-linearity. The upsampling operation is of the form of a transposed convolution, which broadcasts input elements via the filter, thereby producing an output that is larger than the input. The concatenation with the cropped feature map from the contracting path is necessary due to the loss of border pixels at each upsampling path. Additionally, this concatenation improves the reconstruction of the image from the expanding layer by making use of the corresponding learned feature map. At each expanding layer, the number of filters in the 2D convolution layers are halved. Finally, the network has a final 1D convolution followed by

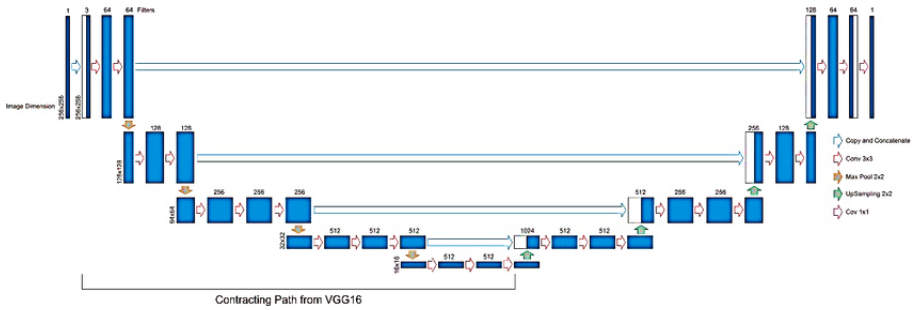

Figure 4. VGG-16 U-Net architecture. Architecture described in depth in methods [11].

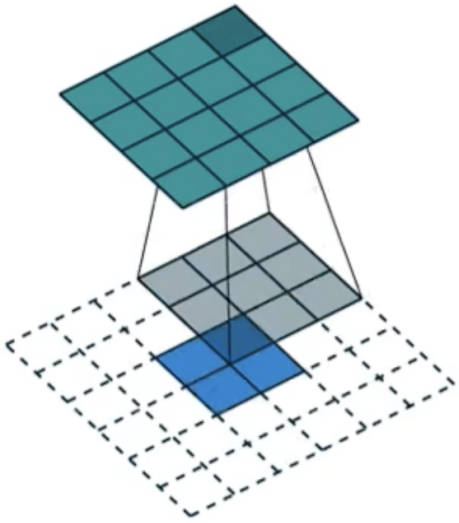

Figure 5. Example of a Transposed Convolution. Notice that the dimension of the original image is expanded after the up-sampling layer. [4]

a softmax nonlinearity in order to map the feature vector to a vector of the number of classes. The architecture is shown in depth in Figure 3.

Additionally, we also decided to expand upon our U-Net model by adding a VGG-16 VGG-19 backbone [11]. When VGGNet was discovered, the smaller filters combined with the deeper network structure outperformed previous models in the ImageNet competition [17]. For example, VGG-16 uses stacks of 3x3 convolution layers which provides the same receptive field as one 7x7 convolutional layer, but with more non-linearities and specificity to the image. In addition, these small kernel sizes lead to fewer parameters that require tuning. In order to incorporate this VGG-Net structure, we added stacks of three 3x3 convolutional layers into our U-Net model. In addition, we also tested out a VGG-19 backbone to our U-Net model by adding stacks of four 3x3 convolutional layers. The architecture describing out VGG-16 U-Net backbone can be found in Figure 4.

We used a code base found on Kaggle to intialize and hypertune our U-Net model [14]. The original model used a dropout rate of 0.2 and a learning rate of 0.001. In order to optimize this model, we added code to augment the data and hypertune the learning rate, dropout rate, probability of data augmentation, and optimizer (described more thoroughly in Section 4: Dataset and Features). This code base included a manual implementation of a U-Net architecture, which allowed us to edit the architecture to add the VGG-16 and VGG-19 backbone.

5. Dataset and Features

The dataset we are working with is the RSNA-ASNR-MICCAI Brain Tumor Segmentation dataset (BraTS'20) [2]. This dataset consists of a multi-institutional routine clinically-acquired MRI scans of glioma. The patients in the datasets had a confirmed glioma diagnosis based on pathology data and methylation patterns within the tumor. The ground truth tumor subregions were manually labeled and identified by one of four raters, which were then approved by expert neuro-radiologists. These identified subregions were included within the training, validation, and testing datasets in order evaluate the tumor segmentations predicted by our model. In total, there are 368 images of brain scans with glioma in BraTS 2020. The images were collected from four different types of MRIs: flair, t1, t1ce, t2. These images are pre-split into 70 percent training, 20 percent validation, and 10 percent testing.

The inputs to the model are MRI multimodal scans in NIfTI format (.nii.gz) from 19 different institutions. For each image is a corresponding mask, which indicates the multi-class "ground truth" as labeled by annotators. This mask contains an integer value for each pixel (0: non-tumor pixel, 1: necrotic and non-enhancing tumor core, 2: peritumoral edema, 4: GD-enhancing tumor).

We normalized our data using the following normalization scheme in order to improve numerical stability and reduce training time:

$$X = (X - X_{min})/(X_{max} - X_{min}).$$

Additionally, we performed data augmentation on our

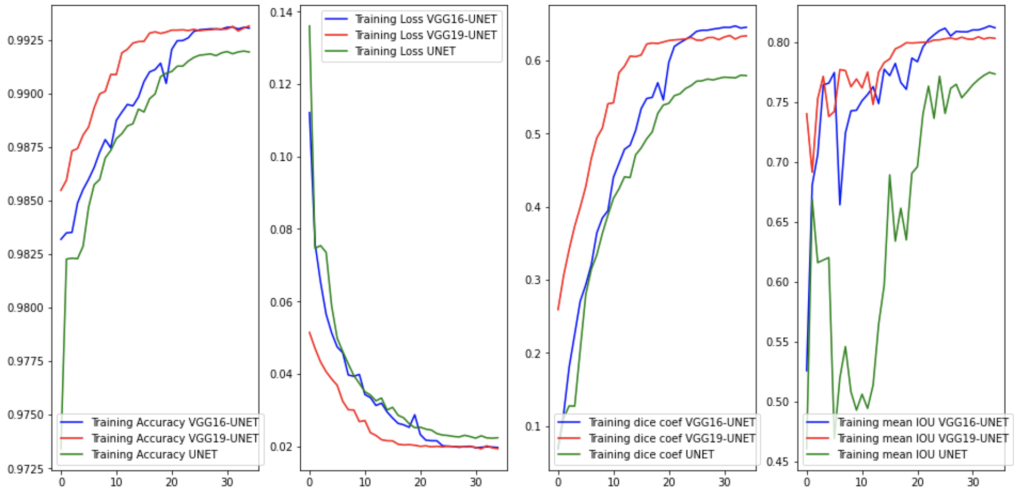

Figure 6. Training Results for U-NET, VGG16 U-NET, and VGG19 U-NET.

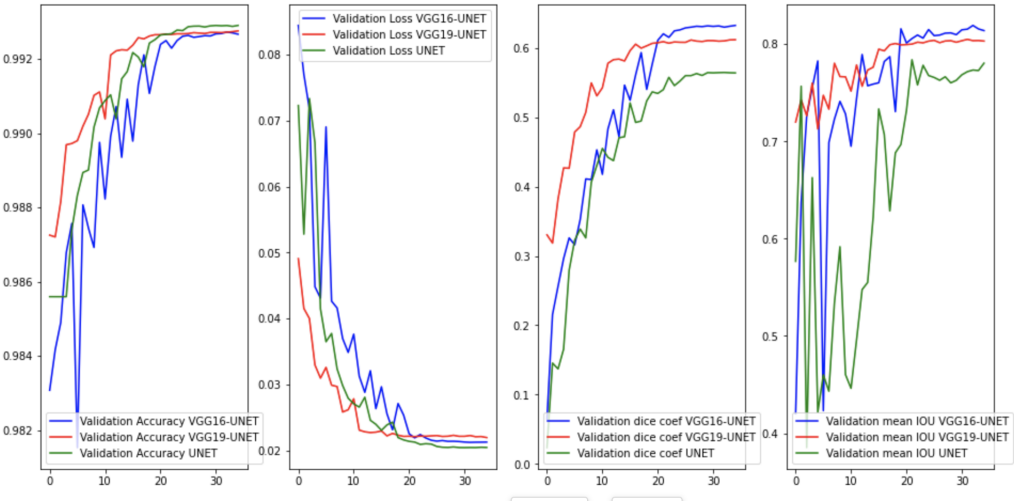

Figure 7. Validation Results for U-NET, VGG16 U-NET, and VGG19 U-NET.

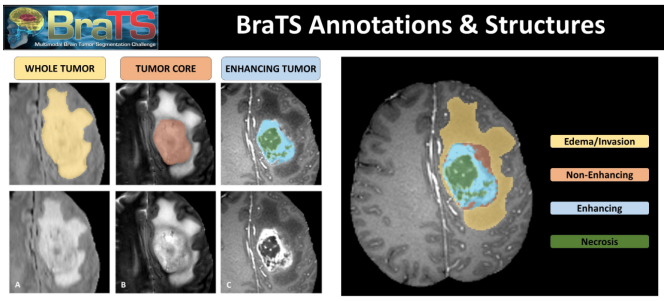

Figure 8. Tumor sub-classes: Edema, Non-enhancing, Enhancing, Necrosis. Each are manually labeled with distinct pixel values.

dataset. With a probability 1-p that was chosen by hyper-parameter tuning, the image was either flipped horizontally, flipped vertically, rotated 90 degrees, had its brightness adjusted from a random factor between -1 and 1, or was transposed. This augmentation helped increase the diversity of data available for our training models.

6. Results

In order to get a baseline of the U-Net model implementation, we ran the original U-Net framework with the given parameters of 0.001 as the learning rate, 0.2 as the dropout, and using the Adam optimizer. In order to improve upon this model, we ran a variety of combinations of different learning rates, dropout rates, data augmentation probabilities and optimizers. The combinations of each parameter ranged from [0.0005, 0.0007, 0.001, 0.005] for learning rate, [0.2, 0.5, 0.7] for dropout rate, and [0.5, 0.3, 0.2] for data augmentation probability. The optimizers we tried were SGD, RMSProp, Adam, and Adagrad. After hyper-tuning our model with these parameter choices, the U-Net

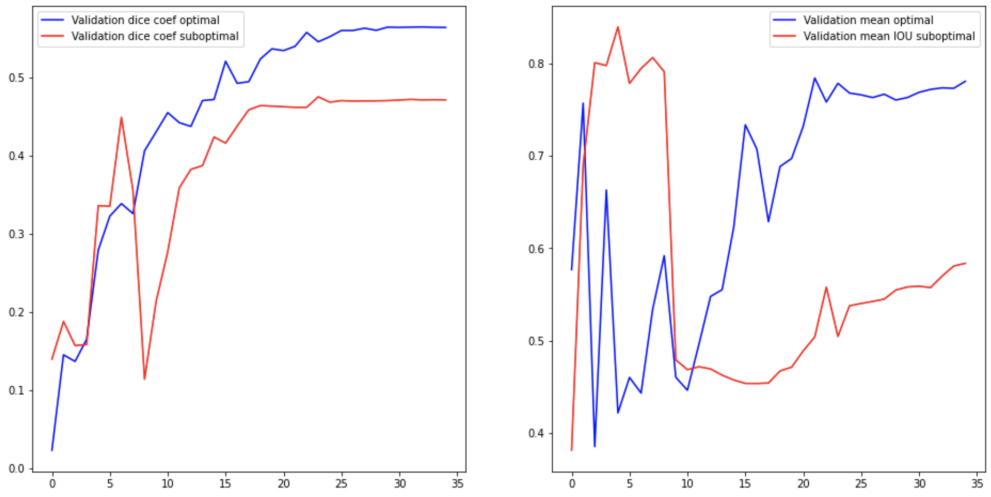

Figure 9. Dice Coefficient and Mean IoU for the optimal vs sub-optimal hyperparameters

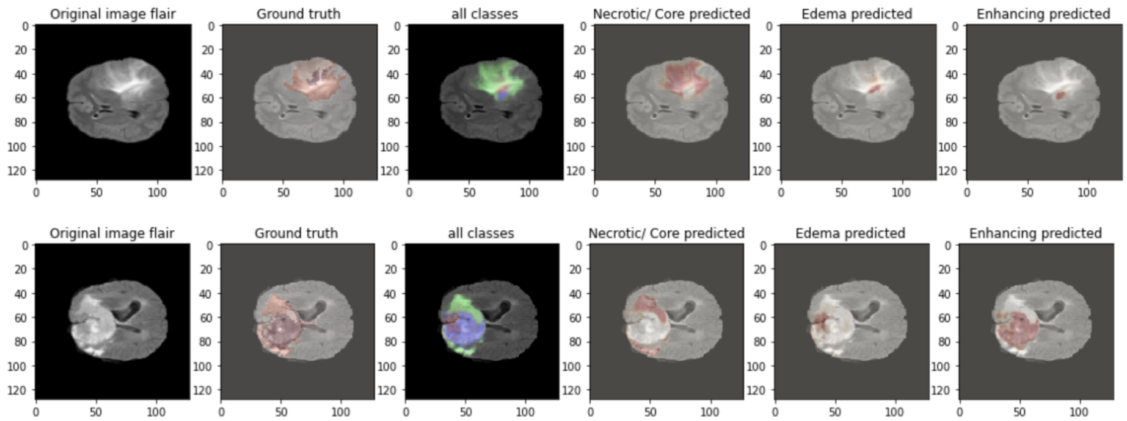

Figure 10. Example segmentation task with VGG16 U-NET and tumor cores. The green area represents the Necrotic Core, the red represents the Edema, and the purple represents the Enhancing. Predicted regions are shown on the latter three right images.

architecture performed best with 0.0007 learning rate, 0.5 dropout, 0.3 data augmentation probability, and the Adam optimizer.

While babysitting the training process, we also noticed that as the loss was decreasing in our model, the large learning rate was causing oscillations in model convergence. Thus, we added the ReduceLROnPlateau from Keras which reduces the learning rate by a factor of 0.2 if the validation loss has not decreased after 2 epochs. Additionally, rather than use smaller mini-batches, we decided to use our entire training set as our batch size for each epoch. Since we had a limited number of training examples (247 brain MRI images), we decided to extract as much information from the training set as possible.

Finally, we also applied the same hyperparameter tuning process to the VGG-16 U-Net and VGG-19 U-Net architecture. These architectures both performed best with 0.0007 learning rate, 0.5 dropout, 0.7 data augmentation probability, and the Adam optimizer.

The metrics we will be using to assess the quality of our model are the Dice Coefficient, the Mean IoU, accuracy, and loss. Please refer to the methods section where we discuss Dice Coefficient and Mean IoU. Accuracy and Loss are based on the number of correctly classified and incorrectly classified pixels. We optimized our hyperparameters with respect to the Dice Coefficient.

Our best model ended up being VGG16 U-NET, which embedded a VGG16 structure into the U-NET architecture. This model achieved a maximum Dice Coefficient of 0.6501 on the test set, which is a significant improvement from the sub-optimal models (shown in table 1 and 2 above). We tried a variety of algorithms, some of which were beneficial and some of which not. After initial improvements with normalization and data augmentation, we tried to incor-

Validation Model Performance				
Model	Accuracy	Loss	Dice Coefficient	Mean IoU
SVM	0.8576	0.1788	0.4367	0.6249
U-Net Suboptimal	0.9878	0.0358	0.4755	0.5835
U-Net Optimal	0.9929	0.0204	0.5648	0.7804
VGG-16 U-Net	0.9927	0.0212	0.6327	0.8191
VGG-19 U-Net	0.9926	0.0219	0.6121	0.8048

Table 1: Validation results for our models. Using dice coefficient as the main evaluation metric, VGG-16 U-NET had the highest dice coefficient of 0.6327 and SVM had the worst performance with a 0.4367.

Test Model Performance				
Model	Accuracy	Loss	Dice Coefficient	Mean IoU
SVM	0.8421	0.1832	0.4521	0.6420
U-Net Suboptimal	0.9922	0.0231	0.4922	0.6419
U-Net Optimal	0.9913	0.0245	0.5723	0.7913
VGG-16 U-Net	0.9910	0.0257	0.6501	0.8127
VGG-19 U-Net	0.9919	0.0232	0.6067	0.8016

Table 2: Test results for our models. Using dice coefficient as the main evaluation metric, VGG-16 U-NET had the highest dice coefficient of 0.6501 and SVM had the worst performance with a 0.4521.

porate Batch Normalization after every 2D Convolutional layer. Batch Normalization did not help in our case, and hindered model performance. Specifically, the Mean IoU score would not achieve higher than than 0.36 and our Dice Score would not achieve a value higher 0.41, indicating that the method was suboptimal. This is likely the case since normalization techniques such as batch normalization tend to wash away semantic information contained in semantic masks [10]. More specifically, it dampens the effect of meaningful pixels in the image.

To further improve upon our basic U-NET, we added a VGG16 and VGG19 structure to the U-NET architecture. This increased the number of convolutional layers within each contracting layer by one and two respectively. This deepened our model, which allowed for more detailed segmentation since each layer increases the complexity of the learned features. Both the VGG16 and VGG19 addition significantly improved our performance relative to the plain U-NET. However, in the validation set, the VGG16 addition performed better than the VGG19 addition in both Dice Coefficient and Mean-IoU metrics. This is likely due to overfitting to the training set, as the VGG19 structure has an additional convolutional layer at each contracting layer, thus deepening the model. Adding a 0.5 dropout after the final contracting layer also improved general model performance, as it reduced overfitting.

7. Conclusions and Future Work

In this project, we explored image segmentation with a variety of deep learning algorithms. We did so in the context of brain images, where we aimed to segment previously annotated brain tumor from MRI scans. In addition to plain tumor segmentation, we also trained our model to segment different structure of the tumor, such as the Necrotic/Core, Edema, and Enhancing region. This study is incredibly useful for the medical field, since manual tumor segmentation is a taxing and time-consuming task for radiologists and technicians. Automating this process is of high value.

We trained a linear SVM, a plain U-NET, and VGG16 U-NET, and a VGG19 U-NET. After optimizing for hyperparameters, we found that the VGG16 U-NET preformed optimally. This is largely due to the beneficial structure of the U-NET, which has proven superior for image segmentation tasks. In a U-NET, the upsampling steps in the expanding path are concatenated with the corresponding (already learned) feature mappings from the contracting path, results in an output that retains structural integrity from the original image [3]. The VGG16 addition results in a deeper network, allowing the model to learn features at various levels of abstraction. Unlike VGG19, it does not appear to overfit to the training set. To further optimize our algorithm, we added data augmentation and 2D spatial dropout, which increased validation performance by expanding upon training examples and regularizing.

In future steps, we plan to investigate different normalization techniques. Batch normalization was a sub-optimal normalization technique since it "washed away" the semantic information of the image. However, authors from [10] propose a spatially-adaptive normalization technique for image segmentation tasks. These spatially-adaptive layers modulate activations so that the network effectively propagates semantic information. Additionally, if we had more time and resources, we would apply elastic deformations as a data augmentation technique, which has proven to be very useful for image segmentation of a variety of tumor MRI

scans [7] [6].

Lastly, future steps include using transfer learning within the VGG16-U-Net model. It is likely that the results from this study can be improved by using a pre-trained model on a similar MRI brain tumor segmentation dataset before fine-tuning to this data. With these additional tools, we could further improve upon an already powerful deep learning model, which has the potential to aid the medical field for years to come.

8. Acknowledgements

We would like to thank @rastislav for the backbone of the U-Net architecture. With his Kaggle project, we were able to add data augmentation, hypertune parameters and build new models to achieve these results [14].

Sasha and Jessica both contributed equally to the work completed in this project. Code writing and model training was performed simultaneously in a group fashion.

References

[1] Brats2020 dataset (training + validation). 1

[2] Multimodal brain tumor segmentation challenge 2020: Data. 1, 4

[3] U-net. 2, 7

[4] Up-sampling with transposed convolution. 4

[5] R. Ayachi and N. Amor. Brain tumor segmentation using support vector machines. 2000. 2

[6] E. Castro, J. Cardoso, and J. Pereira. Elastic deformations for data augmentation in breast cancer mass detection. 2018. 8

[7] M. Cirillo, D. Abramian, and A. Eklund. What is the best data augmentation for 3d brain tumor segmentation? 2021. 8

[8] S. Ghosh, A. Chaki, and K. Santosh. Improved u-net architecture with vgg-16 for brain tumor segmentation. 2021. 2

[9] M. Havaei, H. Larochelle, P. Poulin, and P. Jodoin. Within-brain classification for brain tumor segmentation. 2015. 2

[10] T. Park, M. Liu, T. Wang, and J. Zhu. Semantic image synthesis with spatially-adaptive normalization. 2019. 7

[11] A. Pravitasari, N. Iriawan, U. Nuraini, and D. Rasyid. On comparing optimizer of unet-vgg16 architecture for brain tumor image segmentation. 2022. 4

[12] A. Pravitasari, M. A. N. Iriawan, I. T. Azmi, K. Fithriasari, S. Purnami, and W. Ferriastuti. Unet-vgg16 with transfer learning for mri-based brain tumor segmentation. 2020. 2, 3

[13] N. Punn and S. Agarwal. Modality specific u-net variants for biomedical image segmentation: a survey. 2022. 3

[14] Rastislav. 4, 8

[15] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 1

[16] L. Shen and T. Anderson. Multimodal brain mri tumor segmentation via convolutional neural networks, 2017. 2

[17] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. 2015. 4

[18] V. Zalevskyi. 3