UPSCHOOL MACHINE LEARNING & DEEP LEARNING PROGRAM
IN PARTNERSHIP WITH GOOGLE DEVELOPERS

Low-Grade Glioma Segmentation

Capstone Project

Eda AYDIN

Table of Contents

Table of Figures

A. Business Understanding

This is a capstone project on a real dataset related to segmenting low-grade glioma. This capstone project is included in the *UpSchool Machine Learning & Deep Learning Program in partnership with Google Developers.*

Low-grade glioma is a type of brain tumor that is classified as a grade II and III tumor on the World Health Organization (WHO) grading scale and arises from the supporting cells of the brain called glial cells. They are generally benign but can cause significant neurological symptoms as they grow and put pressure on the surrounding brain tissue. They are most common in adults, but can also occur in children. These tumors are typically treated with surgery, radiation therapy, or a combination of both. In some cases, they may also be observed with close monitoring if the patient is asymptomatic. It is important to note that low-grade gliomas can progress over time to higher-grade tumors, so patients need to undergo regular monitoring to detect any changes in the tumor.

Low-grade glioma segmentation is the process of identifying and separating the tumor tissue from the surrounding healthy tissue in magnetic resonance imaging (MRI) scans of the brain.

Deep learning has proven to be an effective tool for this task, as it allows for the accurate identification of tumor boundaries in MRI scans. Without the use of deep learning, segmentation of low-grade gliomas can be difficult, time-consuming, and subject to inter- and intra-observer variability, as it often requires manual annotation by radiologists. However, the segmentation of low-grade glioma can be challenging due to their infiltrative nature and similarity in appearance to surrounding brain tissue. Additionally, traditional methods may not be as accurate as deep learning, leading to potential misdiagnosis and improper treatment.

As a result, there is a need for accurate and efficient methods for low-grade glioma segmentation to improve patient outcomes. Deep learning-based solutions can improve the efficiency and accuracy of low-grade glioma segmentation, leading to better patient outcomes and cost savings for healthcare providers.

B. Data Understanding

The Brain Magnetic Resonance Imaging (MRI) segmentation dataset is obtained from The Cancer Imaging Archive (TCIA) (Tracy Nolan, 2022). The dataset contains Brain MRI Images together with manual fluid-attenuated inversion recovery (FLAIR)[1] abnormality segmentation masks. They correspond to 110 patients included in The Cancer Genome Atlas (TCGA) lower-grade glioma collection with at least fluid-attenuated inversion recovery (FLAIR) sequence and genomic cluster data available.

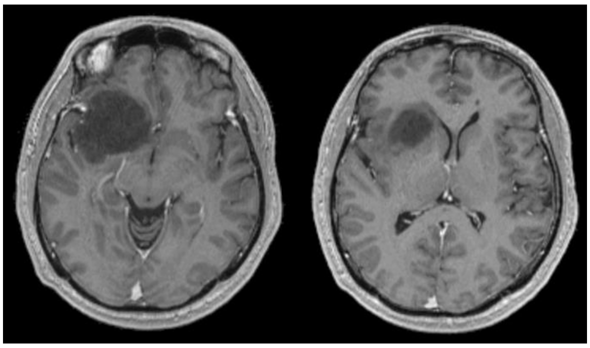


Figure 1 – Low-Grade Glioma Without Segmentation[2]

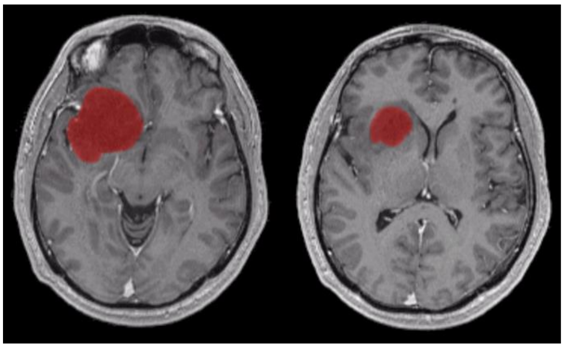


Figure 2 – Low-Grade Glioma With Segmentation[3]

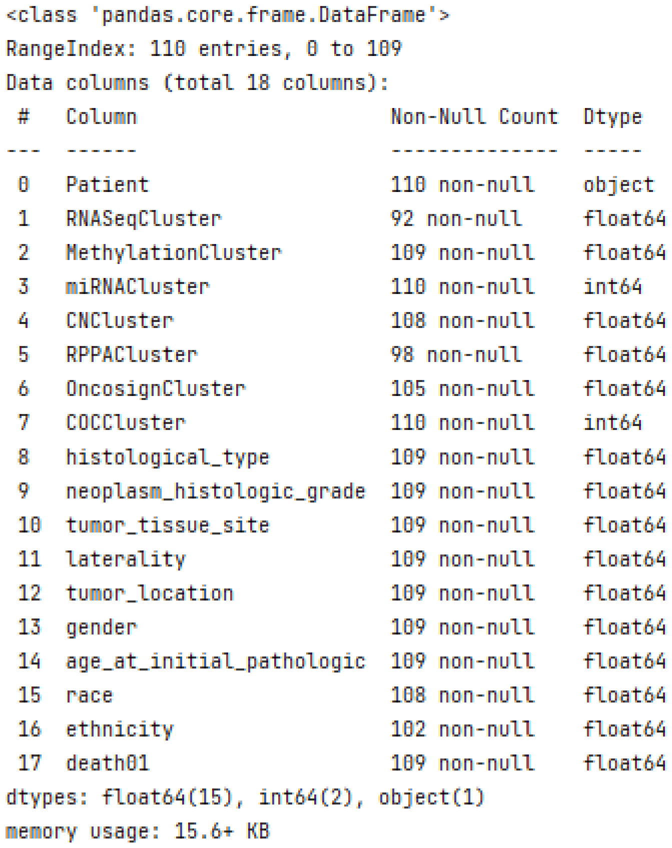
[1] Fluid-attenuated inversion recovery (FLAIR) is an MRI technique that shows areas of tissue T2 prolongation as bright while suppressing (darkening) cerebrospinal fluid (CSF) signal, thus clearly revealing lesions in proximity to CSF, such as cerebral cortical lesions.

[2] This image belongs to a 24-year-old who developed focal seizures affecting the left side of his body.

[3] This image belongs to a 24-year-old who developed focal seizures affecting the left side of his body. The red area indicates the glioma tumor location.

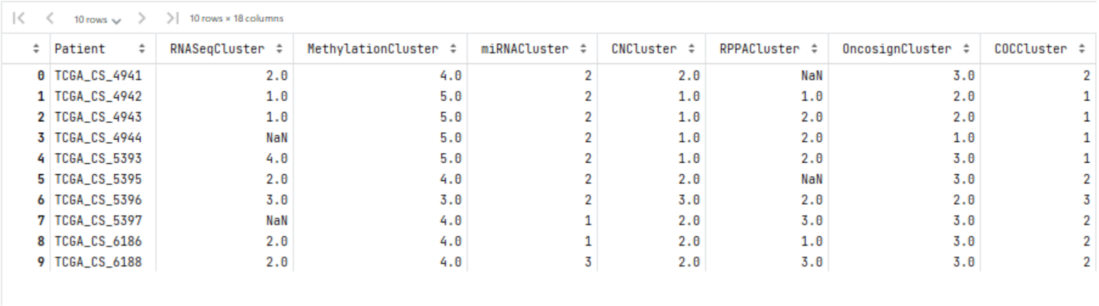
C. Data Preparation

This dataset is a table of 110 rows and 18 columns. Each row represents a patient, and the columns contain various information about the patient such as RNASeqCluster, MethylationCluster, miRNACluster, CNCluster, RPPACluster, OncosignCluster, COCCluster and histological_type, neoplasm_histologic_grade, tumor_tissue_site, laterality, tumor_location, gender, age_at_initial_pathologic, race, ethnicity, death. Each column has a specific data type: object, float64, and int64. The first column "Patient" is of object data type, 15 columns are of float64 data type and 2 columns are of int64 data type. There are missing values in some of the columns which are represented by a "non-null" count in the Data columns section.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110 entries, 0 to 109
Data columns (total 18 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Patient                    110 non-null    object
 1   RNASeqCluster              92 non-null     float64
 2   MethylationCluster         109 non-null    float64
 3   miRNACluster               110 non-null    int64
 4   CNCluster                  108 non-null    float64
 5   RPPACluster                98 non-null     float64
 6   OncosignCluster            105 non-null    float64
 7   COCCluster                 110 non-null    int64
 8   histological_type          109 non-null    float64
 9   neoplasm_histologic_grade  109 non-null    float64
 10  tumor_tissue_site          109 non-null    float64
 11  laterality                 109 non-null    float64
 12  tumor_location             109 non-null    float64
 13  gender                     109 non-null    float64
 14  age_at_initial_pathologic  109 non-null    float64
 15  race                       108 non-null    float64
 16  ethnicity                  102 non-null    float64
 17  death01                    109 non-null    float64
dtypes: float64(15), int64(2), object(1)
memory usage: 15.6+ KB
```

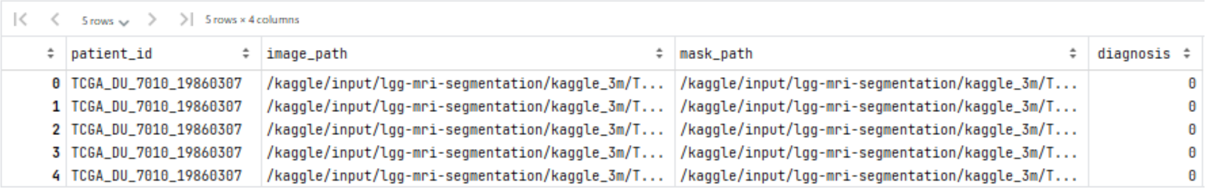
Figure 3- Information about csv dataset



	Patient	RNASeqCluster	MethylationCluster	miRNACluster	CNCluster	RPPACluster	OncosignCluster	COCCluster
0	TCGA_CS_4941	2.0	4.0	2	2.0	NaN	3.0	2
1	TCGA_CS_4942	1.0	5.0	2	1.0	1.0	2.0	1
2	TCGA_CS_4943	1.0	5.0	2	1.0	2.0	2.0	1
3	TCGA_CS_4944	NaN	5.0	2	1.0	2.0	1.0	1
4	TCGA_CS_5393	4.0	5.0	2	1.0	2.0	3.0	1
5	TCGA_CS_5395	2.0	4.0	2	2.0	NaN	3.0	2
6	TCGA_CS_5396	3.0	3.0	2	3.0	2.0	2.0	3
7	TCGA_CS_5397	NaN	4.0	1	2.0	3.0	3.0	2
8	TCGA_CS_6186	2.0	4.0	1	2.0	1.0	3.0	2
9	TCGA_CS_6188	2.0	4.0	3	2.0	3.0	3.0	2

Figure 4 - The head of the csv dataset

Using only the Patient column from the variables in this dataset, the final Dataframe is created with the Brain MRI image and mask files.



	patient_id	image_path	mask_path	diagnosis
0	TCGA_DU_7010_19860307	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	0
1	TCGA_DU_7010_19860307	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	0
2	TCGA_DU_7010_19860307	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	0
3	TCGA_DU_7010_19860307	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	0
4	TCGA_DU_7010_19860307	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	/kaggle/input/lgg-mri-segmentation/kaggle_3m/T...	0

Figure 5 - The final Dataframe to be used in the visualization and modeling part

D. Visualization

The visualization phase takes place in two stages. The first state involves displaying the distribution in different ways. The second stage involves the visualization of Brain MRI Images in different ways.

D.1 Distribution

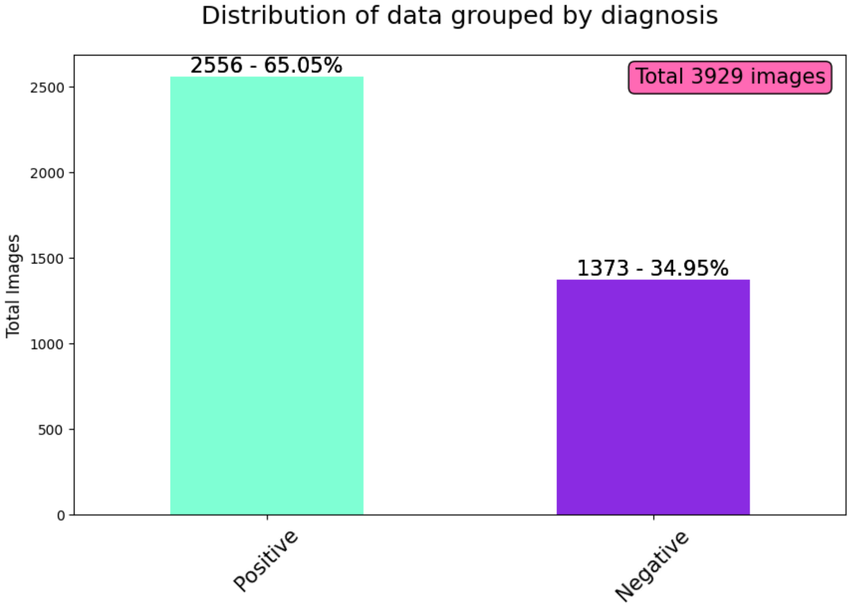


Figure 6 - Distribution of data grouped by diagnosis

This data shows the distribution of images grouped by diagnosis. The data is split into two groups, “positive” and “negative”. The “positive” group contains 2256 images, which represents 65.05% of the total images. The “negative” group contains 1373 images, which represents 34.95% of the total images. In total, there are 3929 images in the dataset.

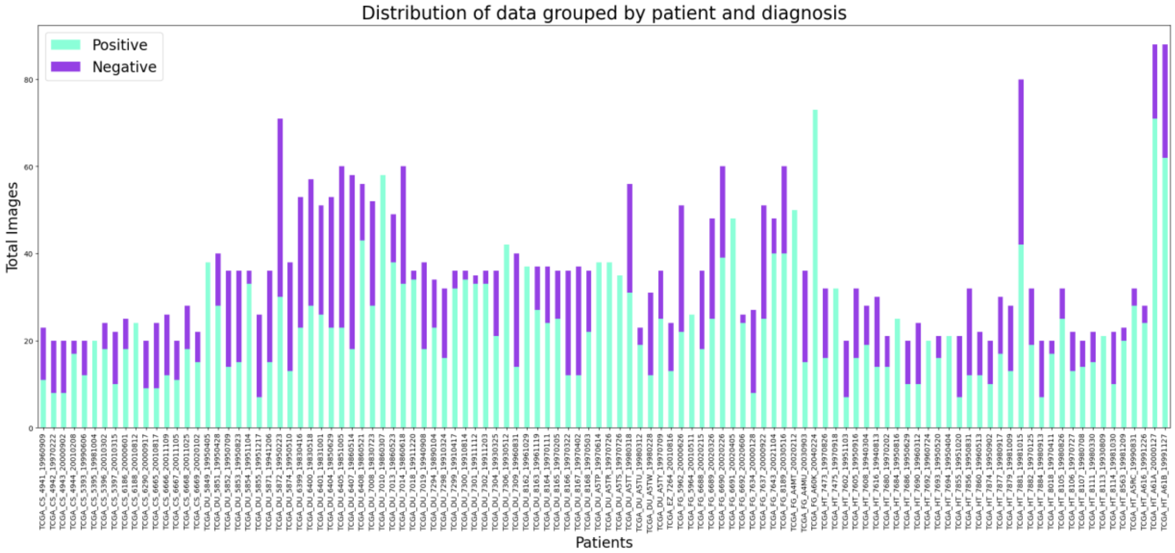


Figure 7 - Distribution of data grouped by patient and diagnosis

Both distributions indicate that there are more positive images than negative images, which could have implications for the performance of any deep learning models trained on this data. For example, if the majority of the data is positive, the model may be more likely to predict a positive diagnosis even when the image is actually negative. Therefore, data augmentation does with these distributions when analyzing or interpreting the results of any deep learning models trained on this data.

D.2 Visualization of Brain MRI Images

Original color and the hot color map[4] used in the visualization of brain MRI images. The main reason why this colormap is important is that it can effectively highlight different tissue types in the brain based on their MRI intensities. The hot color maps low-intensity values to dark colors (black) and high-intensity values to bright colors (red, yellow, and white). This helps to distinguish different structures in the brain such as grey matter, white matter, and cerebrospinal fluid.

Additionally, the hot colormap is also useful in differentiating between healthy and abnormal tissue, such as tumors. Tumors often have different intensity values compared to the surrounding healthy tissue and using a colormap like hot can help to highlight these differences and make them more visible. This can be helpful for radiologists and physicians when analyzing the images and can be useful in determining the diagnosis and treatment of the patient.

It is important to note that the choice of colormap is not only important for visualizing the data but also for the analysis of the data, for example in brain tumor segmentation tasks, the choice of colormap will be important to define the threshold values for the segmentation algorithm.

[4] Hot color map is a color map from matplotlib library that is commonly used to visualize brain MRI images.

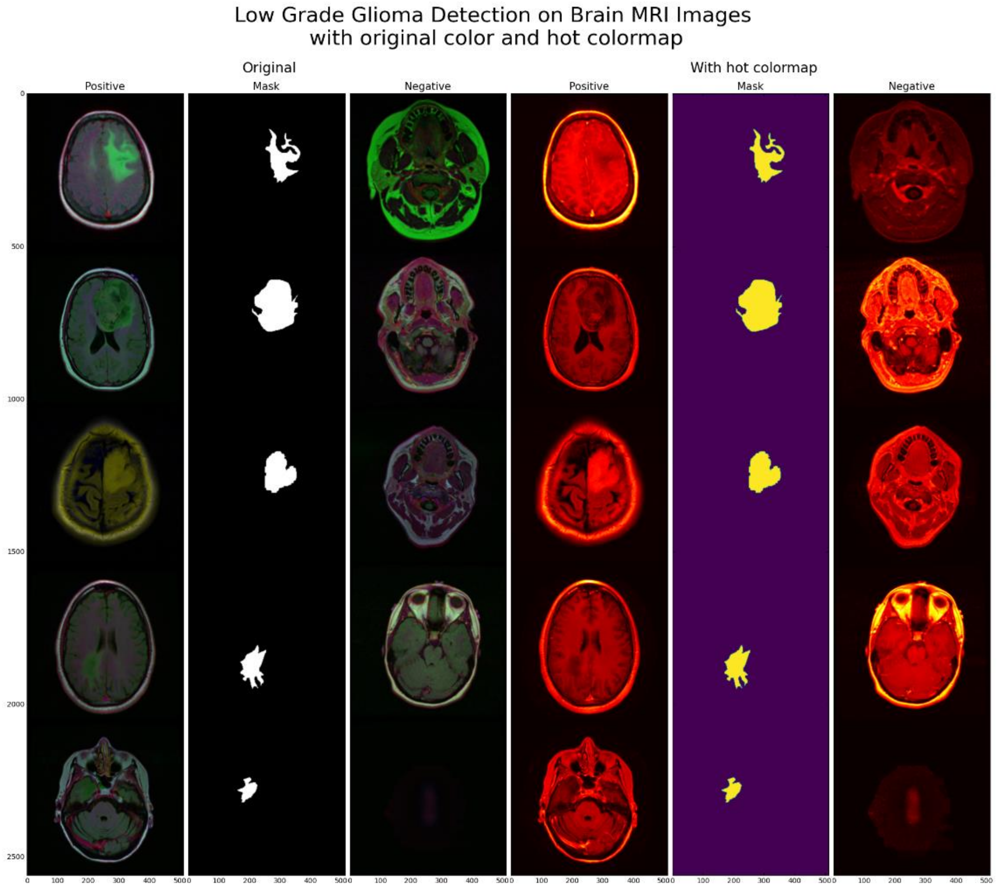


Figure 8 – Low-Grade Glioma Detection on Brain MRI Images with original color and hot colormap

This diagram shows the representation of different images, positive and negative, under two different color scales. (Original color and hot colormap) Positive images using a hot colormap show where the tumor is more black or more white.

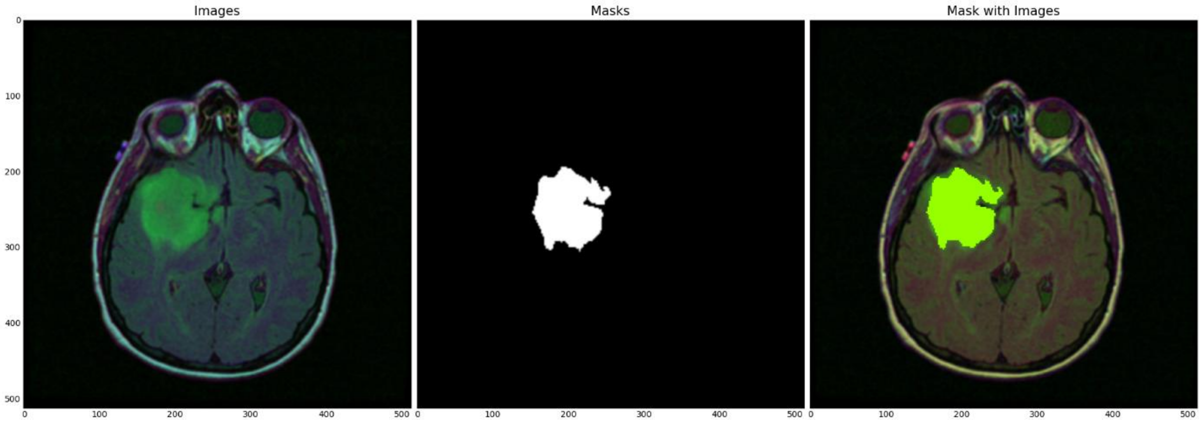


Figure 9 - Tumor location is shown as segmented on a Brain MRI image

In more detail, this diagram shows both a brain MRI image, a segmented mask, and the segmented mask on a brain MRI image.

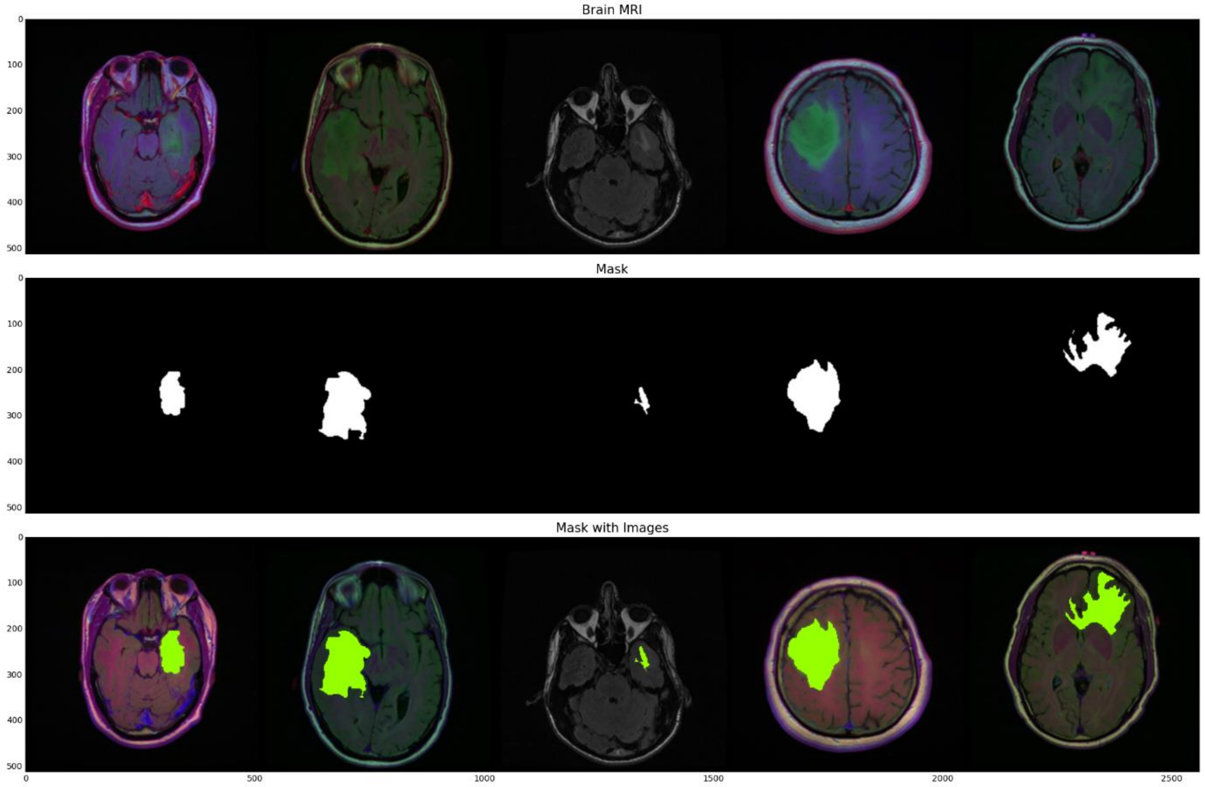


Figure 10 - Tumor location is shown as segmented on multiple Brain MRI images

In more detail, this diagram shows multiple brain MRI images, segmented masks, and segmented masks on multiple brain MRI images.

E. Data Augmentation

In the data augmentation part, the Albumentations library will be used because of its wide range of image segmentation techniques that can be applied to medical images. The library provides specialized augmentations for medical images such as elastic deformation, brightness, contrast changes, and random rotation, scaling, and flipping which are important to simulate the variability that can occur in medical images. This can help to improve the robustness and generalization of the model by providing more diverse training data.

Additionally, Albumentations allows for the easy composition of multiple augmentations together to create complex data augmentation pipelines, which can be useful for medical image segmentation tasks where multiple augmentations may be needed to account for variations in images.

In more detail, apart from normal augmentation techniques, one of the 3 different augmentation techniques are applied. These are elastic deformation, grid distortion, and optical distortion.

Elastic Deformation is a technique that used a random displacement field to deform the images. This can make the model robust to small variations in the input data.

Grid Distortion is a technique that wraps the image by applying a random grid of points. This can make the model more robust to small variations in the input data.

Optical Distortion is a technique that simulates the effects of a camera lens by applying random distortions to the image. This can make the model more robust to small variations in the input data, particularly for image classification tasks.

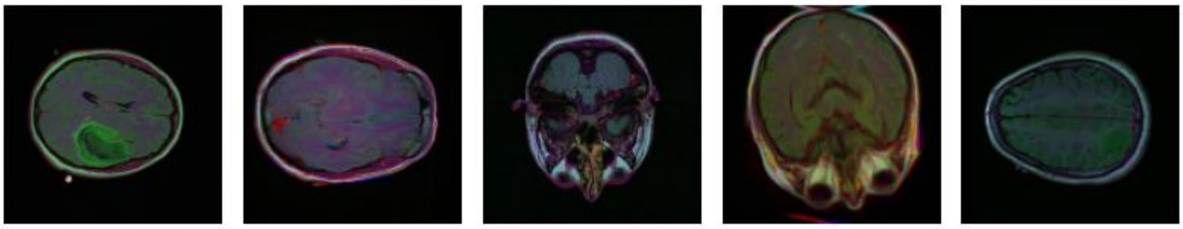


Figure 11 - Augmented Brain MRI Images

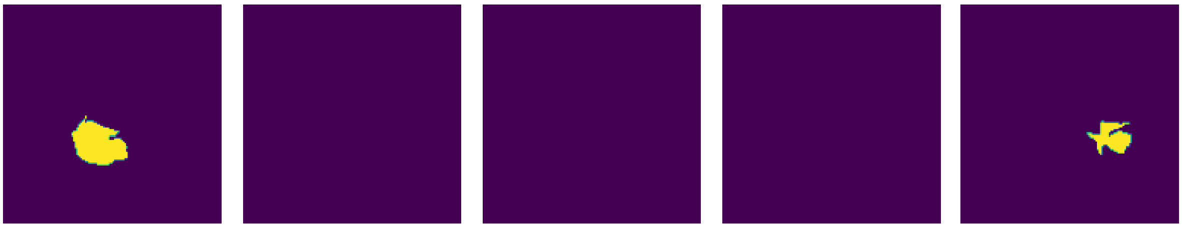


Figure 12 - Augmented Mask Images

F. Modeling

F.1 Model Information and Decision

Several neural network models can be used for the segmentation of low-grade glioma, each with their strengths and weakness. Some popular models that have been used for this task include:

1. *U-Net*: It is a specific type of FCN that is designed for biomedical image segmentation. It is composed of an encoder and a decoder, which are trained to learn features from the input image and then use these features to predict a segmentation mask. It has been widely used in brain tumor segmentation tasks. The encoder is composed of a series of convolutional layers that extract features from the input image, and the decoder is composed of a series of up sampling layers that increase the spatial resolution of the features.
2. *FPN (Feature Pyramid Network)*: It is a type of deep neural network architecture that is designed for object detection and semantic segmentation tasks. FPNs are built on top of a CNN, and they are used to extract features at multiple scales. They work by creating a pyramid of feature maps, where each level of the pyramid corresponds to a different scale.
3. *ResUNet*: ResNet is a type of CNN that uses residual connections between layers to allow the network to learn deeper and more complex representations of the input data. U-Net is a type of CNN that is designed for biomedical image segmentation. It is composed of an encoder and a decoder, which are trained to learn features from the input image and then use these features to predict a segmentation mask. ResUNet combines the encoder-decoder architecture of U-Net with the residual connections of ResNet, this allows the network to learn more robust features from the input images and improve the performance of the segmentation task.

F.2 Hardware and Software Requirements

PyTorch is used for this project instead of Keras, and TensorFlow. There are several reasons for using PyTorch:

- PyTorch has a more intuitive and flexible API, which makes it easier to work with for complex tasks such as segmentation and Albumentations library. This allows for more experimentation and iteration in the development process.
- PyTorch has built-in support for dynamic computational graphs, which can be useful for tasks such as segmentation where the model needs to adapt to different input shapes and sizes.
- PyTorch allows for easy debugging, since the forward pass is defined by a single function, and the backward pass is defined by the auto grad system.

F.3 Building Vanilla U-Net Architecture

The U-Net architecture is composed of an encoder network and a decoder network, which are connected by a set of "skip connections."

The encoder network is made up of a series of convolutional and max pooling layers that progressively reduce the spatial resolution of the input image while increasing the number of feature maps. This allows the network to extract increasingly complex and abstract features from the image.

The decoder network is made up of a series of up sampling layers and convolutional layers that restore the spatial resolution of the feature maps while decreasing the number of feature maps. The output of the decoder network is a segmentation mask that corresponds to the input image.

The "skip connections" concatenate feature maps from corresponding layers in the encoder and decoder networks, allowing the network to make use of both high- and low-level information from the input image.

The diagram of U-Net architecture would typically show the input image at the top, and the output segmentation mask at the bottom. The encoder network would be shown on the left side of the diagram, with the layers arranged vertically and labeled with the number of feature maps and the size of the kernel for the convolutional layers. The decoder network would be shown on the right side of the diagram, with the layers arranged vertically and labeled with the number of feature maps and the size of the kernel for the convolutional layers. The "skip connections" would be shown as arrows connecting corresponding layers in the encoder and decoder networks.

In this study, U-Net-35 is used, consisting of 15 convolutional, 14 rectified linear units (ReLU), three max-pooling, and three up sampling layers.

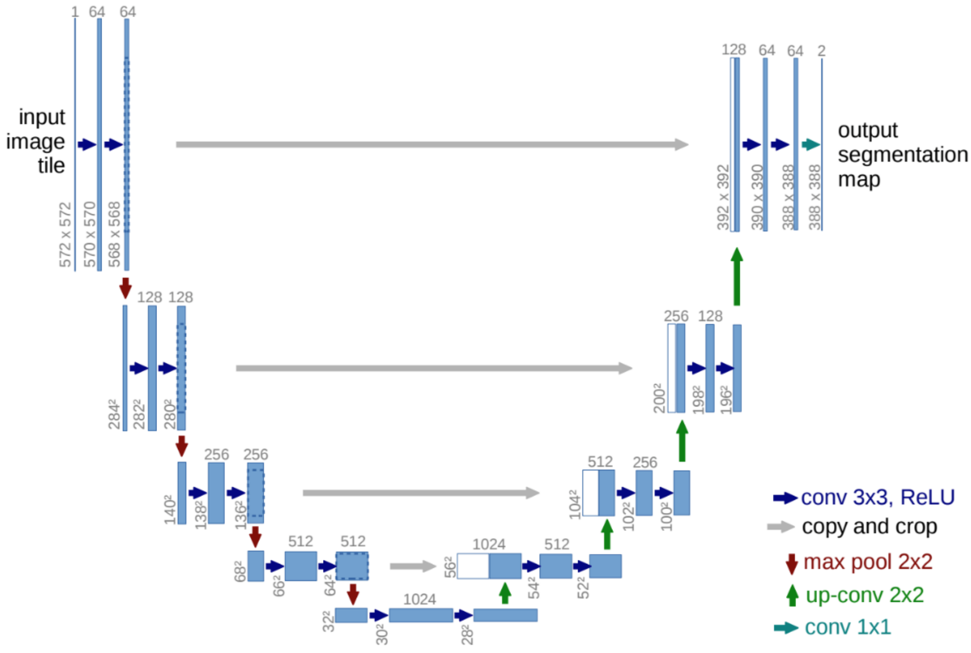


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Figure 13 – U-Net Architecture from U-Net: Convolutional Networks for Biomedical Image Segmentation Article

F.4 Building Feature Pyramid Network (FPN) Architecture

FPN, or Feature Pyramid Network, is a type of neural network architecture for object detection in computer vision. It is built on top of a backbone CNN, such as ResNet or ResNeXt, and consists of several layers:

- The bottom-up pathway starts with the output feature maps of the last convolutional layer of the backbone CNN. These feature maps are then passed through a series of convolutional layers, known as the "lateral layers," that reduce the spatial resolution but increase the number of feature maps.
- The top-down pathway starts with the output of the lateral layers and uses a series of transposed convolutional layers, known as the "up sampling layers," to increase the spatial resolution of the feature maps.
- The final layers of the FPN combine the feature maps from the bottom-up pathway and the top-down pathway to form the final feature pyramid.
- Finally, The final feature pyramid is then passed to a head layer for classification and bounding box regression task.

The FPN architecture aims to combine the advantages of both high-level semantic information and low-level spatial information to improve object detection performance.

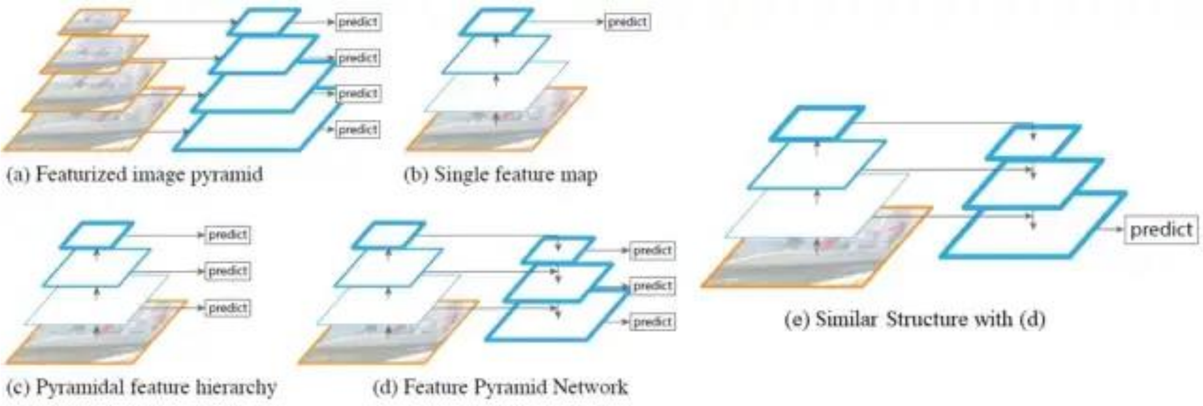

predict
(a) Featurized image pyramid
(b) Single feature map
(c) Pyramidal feature hierarchy
(d) Feature Pyramid Network
(e) Similar Structure with (d)

Figure 14 - Feature Pyramid Network Architecture

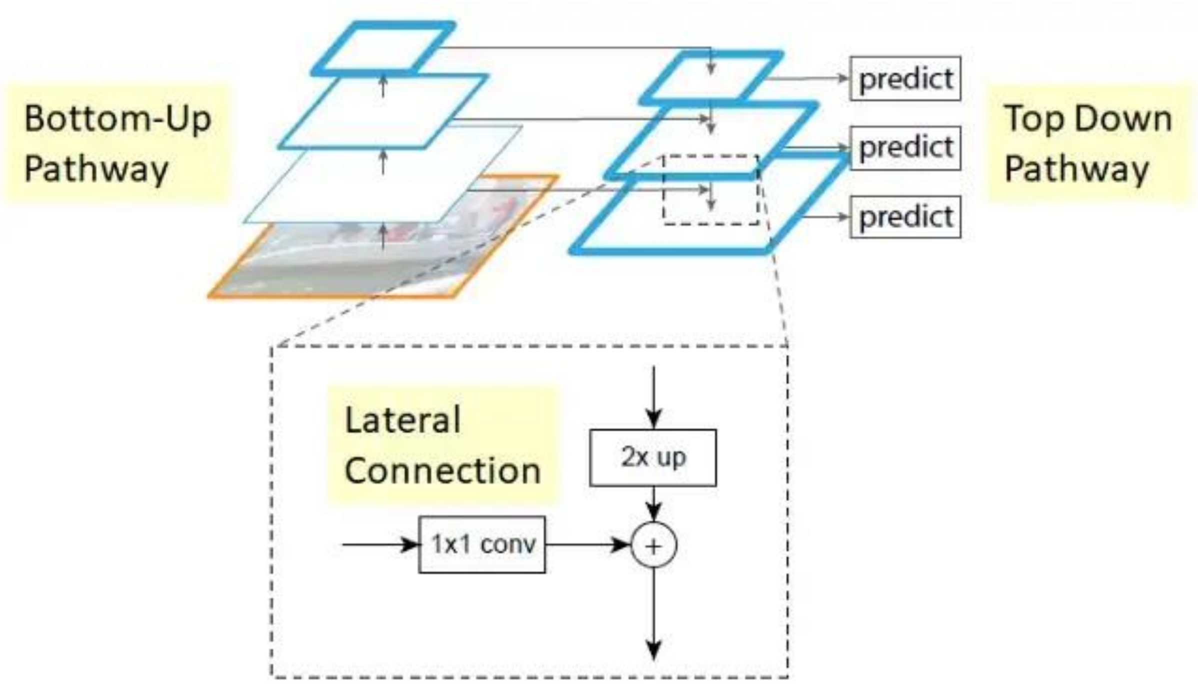


Figure 15 - Explanation of Bottom-Up- Top Down and Lateral Layers

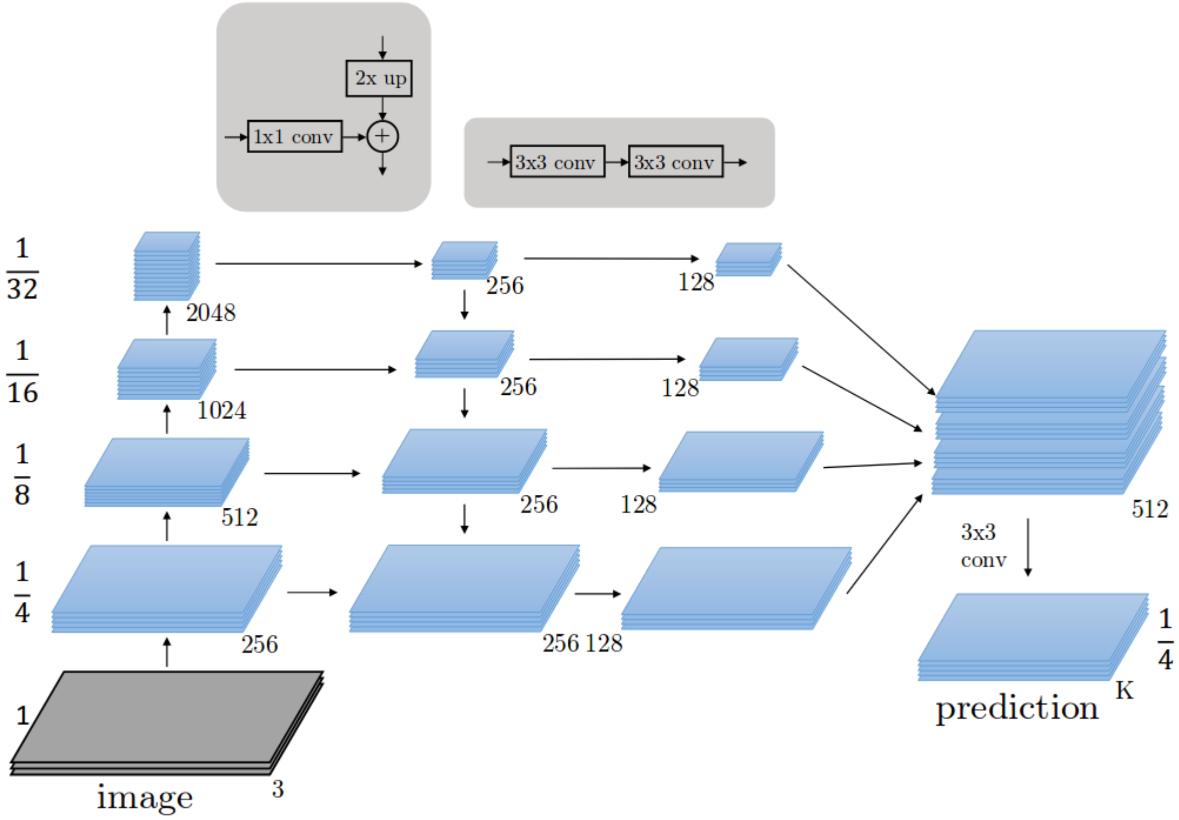


Figure 16 - Merging the FPN last layers with UNet architecture

The architecture of the Feature Pyramid Network (FPN) architecture using the PyTorch library is divided into several parts. It takes an input image and applies the *bottom-up layers* to create a pyramid of feature maps with decreasing spatial resolutions. The *top-down layer* is then applied to the highest-resolution feature map, and the feature maps from the bottom-up layers are combined with the feature maps from the top-down layer using the *lateral layers*. The *smooth layers* are then applied to further refine the feature maps, and the segmentation block layers are applied to generate the final segmentation maps. The *last layer* is used to generate final predictions by converting the feature maps from the segmentation block into a single output map.

In this study FPN is used, consisting of 18 convolutional, 10 rectified linear units (ReLU), five max-pooling, five bottom-up layers, one top-down layer, three up sampling addition layers, three smooth layers, and one up sampling layer.

F.5 Building U-Net with ResNext50 Backbone Architecture

A U-Net with a ResNeXt50 backbone is an implementation of the U-Net architecture, where the encoder part of the network is a ResNeXt50 model. ResNeXt50 is a deep CNN architecture that was introduced in 2016 and has been shown to achieve state-of-the-art performance on various computer vision tasks, such as object detection and image classification. By using a pre-trained ResNeXt50 model as the encoder, the U-Net model can leverage the rich feature representations learned by the ResNeXt50 model, which allows it to better segment objects in the input image.

In this study, the ResNext50-32x4d model is used for the pre-trained model, and five down-sampling layers, four up-sampling layers, two convolutional layers, and one rectified linear unit (ReLU) are added to this model.

F.6 Segmentation Quality Metric

$$\text{Dice coefficient} = \frac{2 \times \text{Intersection of predicted and target}}{\text{Predicted pixels} + \text{Target pixels}}$$

Dice Coefficient Metric is used as the segmentation quality metric. This metric measures the similarity between the predicted segmentation and the ground truth segmentation. It ranges from 0 (no overlap) to 1 (perfect overlap).

F.7 Segmentation Loss

Dice Coefficient Loss is used as the segmentation loss. This is a measure of how well the output of a segmentation model matches the ground truth. The BCE (binary cross-entropy) loss function from PyTorch' s nn module is used to calculate the binary cross-entropy loss between the input and target tensors. This loss is calculated as:

$$-\left(target * log(input) + (1 - target) * log(1 - input)\right)$$

G. Evaluation

G.1 Evaluation of the Model Architectures on Training and Validation Data

It is concluded that the mean DICE values on the validation data of both U-Net and FPN model architectures exceeded 80%. It is concluded that the architecture of the ResNext-50 model exceeded 90%.

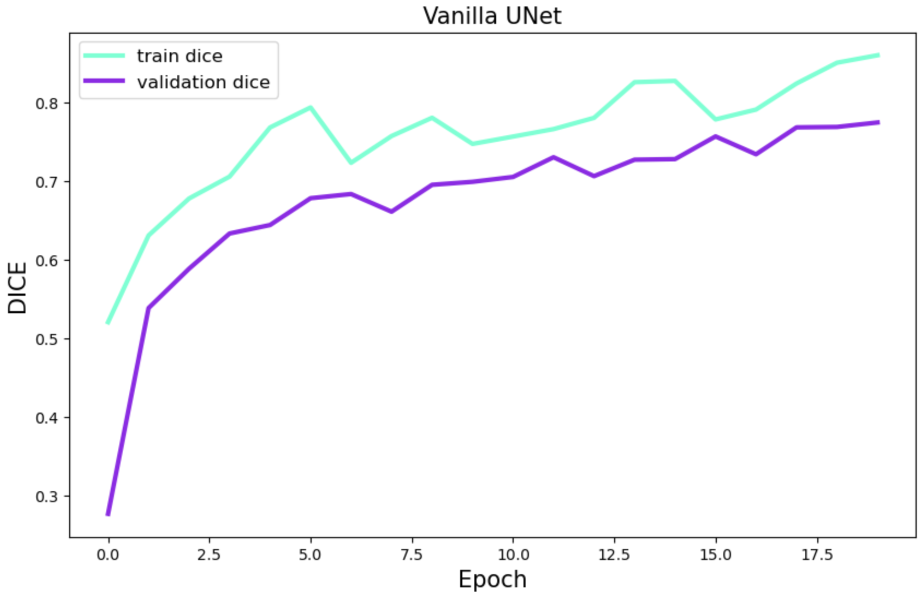


Figure 17 - Epoch vs. DICE with Vanilla U-Net Architecture

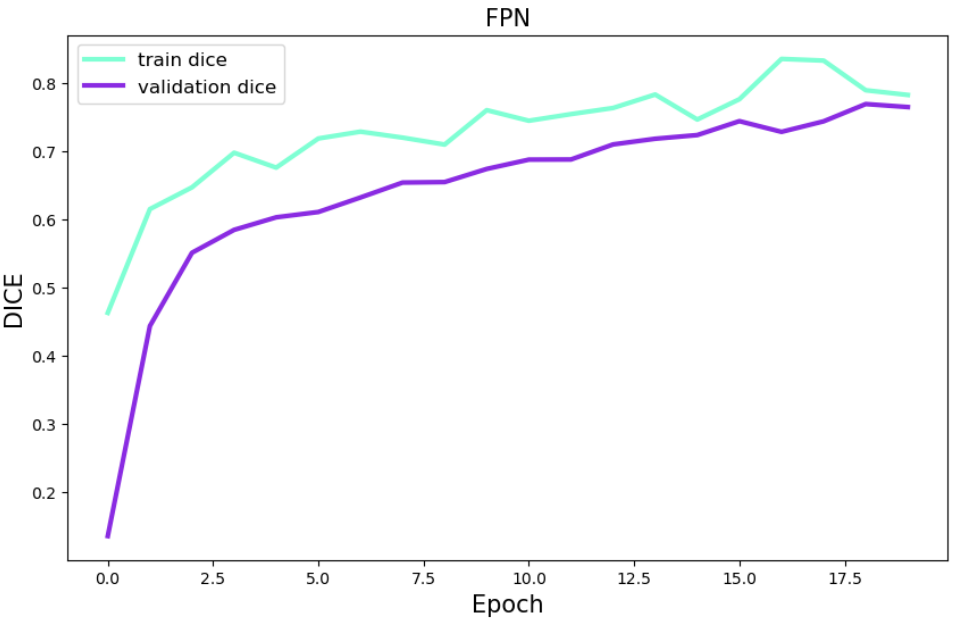


Figure 18 - Epoch vs. DICE with FPN Architecture

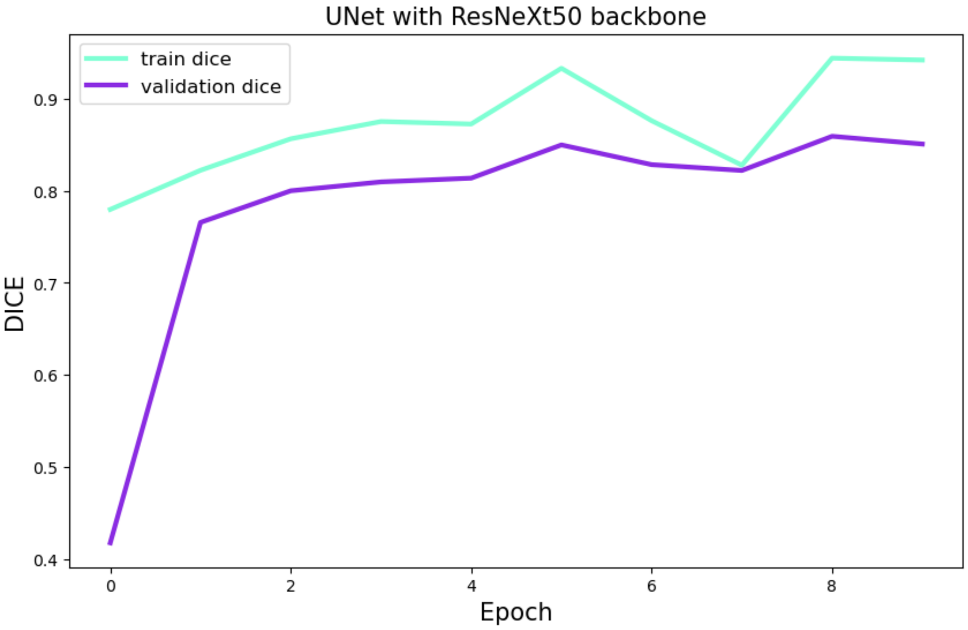


Figure 19 - Epoch vs. DICE with U-Net- ResNeXt50 Architecture

G.2 Evaluation of the Model Architectures on Test Data

It was concluded that the mean DICE value of the U-Net model architecture on the test data was 83%, the mean DICE value of the FPN model architecture on the test data was 78%, and the mean DICE value of the U-Net with ResNeXt50 Backbone model architecture on the test data was 89%.

G.3 Evaluation of the Random Test Sample

In the data augmentation part, it is mentioned that different threshold values play an important role in segmenting the tumor. Models built on this are tried to predict the segmentation of the tumor on a random test sample, both without applying a threshold and within a threshold range. As a result, the same results are obtained in both.

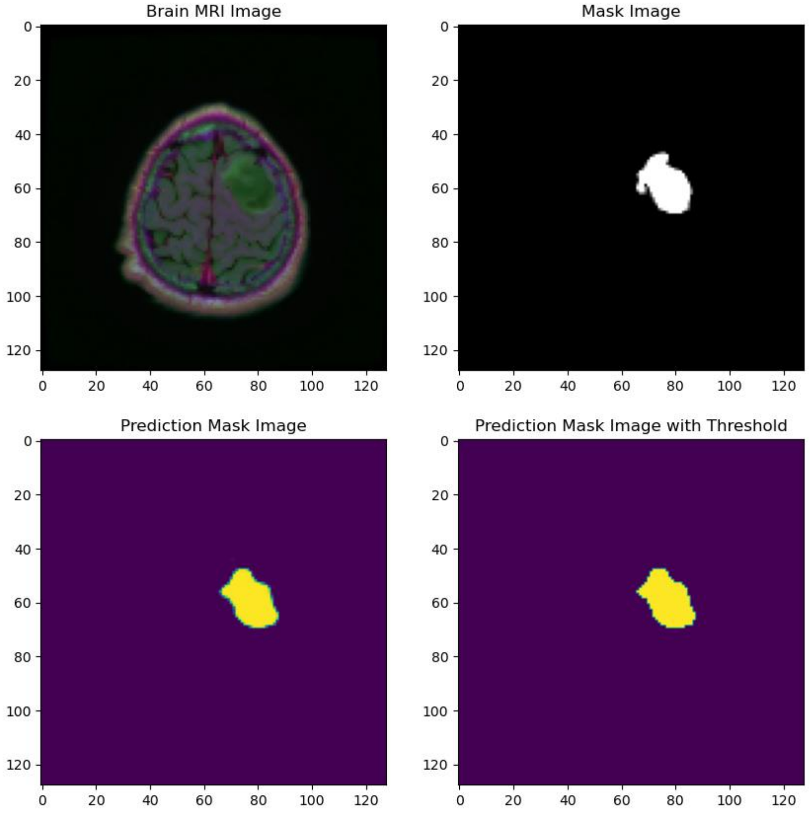


Figure 20 - Prediction Mask Image without Threshold and with Threshold

G.4 Evaluation of the Prediction and Ground Truth Masks on the Brain MRI Images

The results of Prediction and Ground Truth Masks on Brain MR Images are shown on different models. As can be seen from the pictures, the ResNeXt-50 model has been more successful.

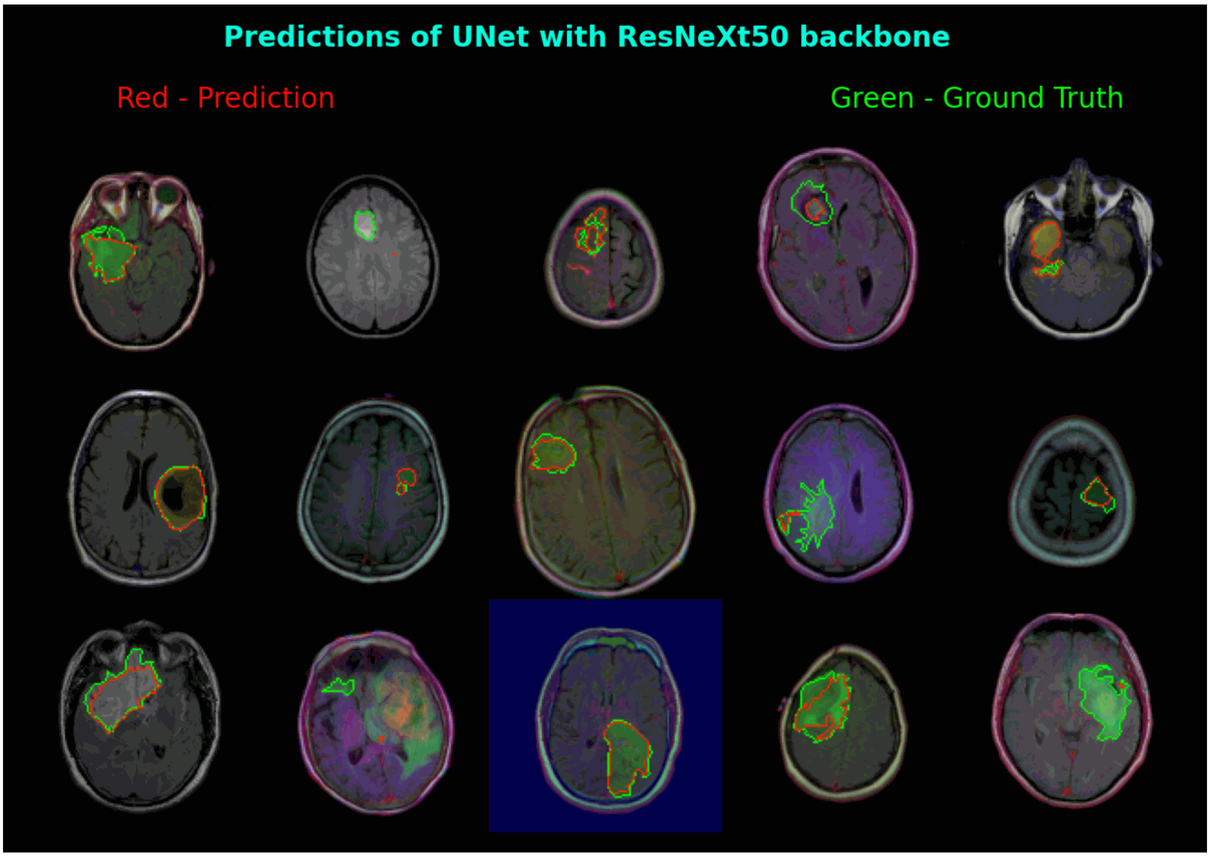


Figure 21 - Prediction and Ground Truth Masks on the Brain MR Images with ResNeXt50 Backbone

References

(n.d.). Retrieved from Albumentations Documentations: https://albumentations.ai/docs/

A Unified Architecture for Instance and Semantic Segmentation. (n.d.). Retrieved from http://presentations.cocodataset.org/COCO17-Stuff-FAIR.pdf

Bakshi R, Ariyaratana S, Benedict RHB, Jacobs L. Fluid-Attenuated Inversion Recovery Magnetic Resonance Imaging Detects Cortical and Juxtacortical Multiple Sclerosis Lesions. Arch Neurol. 2001;58(5):742–748. doi:10.1001/archneur.58.5.742

Bonhart. (2020, May 2). *Brain MRI | Data Visualization | UNet | FPN*. Retrieved from Kaggle: https://www.kaggle.com/code/bonhart/brain-mri-data-visualization-unet-fpn

Buda, M., Saha, A., & Mazurowski, M. A. (2019). Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm. Computers in Biology and Medicine, 109, 218–225. https://doi.org/10.1016/j.compbiomed.2019.05.002

Brain Tumor Segmentation. (n.d.). Retrieved from Papers with Code: https://paperswithcode.com/task/brain-tumor-segmentation

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: Fast and Flexible Image Augmentations. Information, 11(2), 125. https://doi.org/10.3390/info11020125

Forst, D. A., Nahed, B. V., Loeffler, J. S., & Batchelor, T. T. (2014). Low-grade gliomas. The oncologist, 19(4), 403–413. https://doi.org/10.1634/theoncologist.2013-03450.

Gupta, A. (2021, February 6). *Brain MRI Detection | Segmentation | ResUNet*. Retrieved from Kaggle: https://www.kaggle.com/code/anantgupt/brain-mri-detection-segmentation-resunet

How to change the color of an image using a mask? (2020, July 4). Retrieved from Stack Overflow: https://stackoverflow.com/questions/62891917/how-to-change-the-colour-of-an-image-using-a-mask

Hui, J. (2020, April 30). *Understanding Feature Pyramid Networks for object detection (FPN)*. Retrieved from Medium: https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c

Limam, M. (2021, October 18). *Brain MRI Segmentation*. Retrieved from Kaggle: https://www.kaggle.com/code/mahmoudlimam/brain-mri-segmentation

Mazurowski, M. (n.d.). *Deep learning based skull stripping and FLAIR abnormality segmentation in brain MRI using U-Net*. Retrieved from GitHub: https://github.com/MaciejMazurowski/brain-segmentation

M. B. Khan, P. S. Saha, and A. D. Roy, "Automatic Segmentation and Shape, Texture-based Analysis of Glioma Using Fully Convolutional Network," 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), Rajshahi, Bangladesh, 2021, pp. 1-6, doi: 10.1109/ACMI53878.2021.9528282.

Nalepa, J., Marcinkiewicz, M., & Kawulok, M. (2019). Data Augmentation for Brain-Tumor Segmentation: A Review. Frontiers in Computational Neuroscience, 13. https://doi.org/10.3389/fncom.2019.00083

R., R. (2021, February 22). *BrainMRI|UNet|FPN|ResNeXt50*. Retrieved from Kaggle: https://www.kaggle.com/code/raviyadav2398/brainmri-unet-fpn-resnext50

Team, K. (2021, December 12). *Carvana Image Masking Challenge–1st Place Winner's Interview*. Retrieved from Medium: https://medium.com/kaggle-blog/carvana-image-masking-challenge-1st-place-winners-interview-78fcc5c887a8

Thada, V., & Jaglan, V. (2013). Comparison of Jaccard, Dice, Cosine Similarity Coefficient To Find Best Fitness Value for Web Retrieved Documents Using Genetic Algorithm.

Tracy Nolan, E. B. (2022, September 14). *LGG-1p19qDeletion*. Retrieved from Cancer Imaging Archive: https://wiki.cancerimagingarchive.net/display/Public/LGG-1p19qDeletion#2578904280117571858341b9a6da8a7677b95ed4

T. R. E. Armstrong, P. Manimegalai, A. Abinath, and D. Pamela, "Brain tumor image segmentation using Deep learning," 2022 6th International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, India, 2022, pp. 48-52, doi: 10.1109/ICDCS54290.2022.9780707.

Tsang, S. (2021, December 7). *Review: FPN — Feature Pyramid Network (Object Detection)*. Retrieved from Medium: https://towardsdatascience.com/review-fpn-feature-pyramid-network-object-detection-262fc7482610

U-Net Explained. (n.d.). Retrieved from Papers with Code: https://paperswithcode.com/method/u-net