Brain Tumor MRI Segmentation

Sanjeev Narasimhan

Introduction

This report aims to provide an introductory segmentation of brain MRI scans to isolate the location of brain tumors. Segmentation was done using a U-Net network to segment the MRI scans and generate the predicted tumor mask. Two different datasets were used to carry out the segmentation task. The first dataset was used to train and fine tune the model, and the second was used as a true test dataset.

A brain tumor is a collection, or mass, of abnormal cells in your brain. Your skull, which encloses your brain, is very rigid. Any growth inside such a restricted space can cause problems. Brain tumors can be cancerous (malignant) or noncancerous (benign). When benign or malignant tumors grow, they can cause the pressure inside your skull to increase. This can cause brain damage, and it can be life-threatening.

Early detection and classification of brain tumors is an important research domain in the field of medical imaging and accordingly helps in selecting the most convenient treatment method to save a patient's life.

Overview of Data

Dataset 1

Dataset Description:

- Brain MRI images together with manual FLAIR abnormality segmentation masks. The images were obtained from The Cancer Imaging Archive (TCIA).
- They correspond to 110 patients included in The Cancer Genome Atlas (TCGA) lower-grade glioma collection
- 7860 .tif files: multiple images per MRI scan

Train/ Test Split: The dataset was split into 80% train and 20% test

- 80% Train –> 1415 files
- 20% Test –> 354 files

Mask: for each MRI image was an associated mask that displayed the shape and location of the tumor in the respective image

Dataset 2

Dataset Description:

- This dataset contains 7023 .jpg images of human brain MRI images which are classified into 4 classes: glioma, meningioma, no tumor, pituitary

Glioma:

- Glioma is a common type of tumor originating in the brain, but it can sometimes be found in the spinal cord. About 33% of all brain tumors are gliomas. These tumors arise from the glial cells that surround and support neurons. There are several types of glial cells, hence there are many types of gliomas, including: Astrocytomas, Oligodendrogliomas, and Glioblastomas. Depending on the type of cells that are forming the glioma and their genetic mutations, those tumors can be more or less aggressive. As such, a genetic study of the tumor is often performed to better understand how it may behave.

Meningioma:

- Meningioma is the most common primary brain tumor, accounting for more than 30% of all brain tumors. Meningiomas originate in the meninges, the outer three layers of tissue that cover and protect the brain just under the skull. About 85% of meningiomas are noncancerous, slow-growing tumors. Almost all meningiomas are considered benign, but some meningiomas can be persistent and come back after treatment.

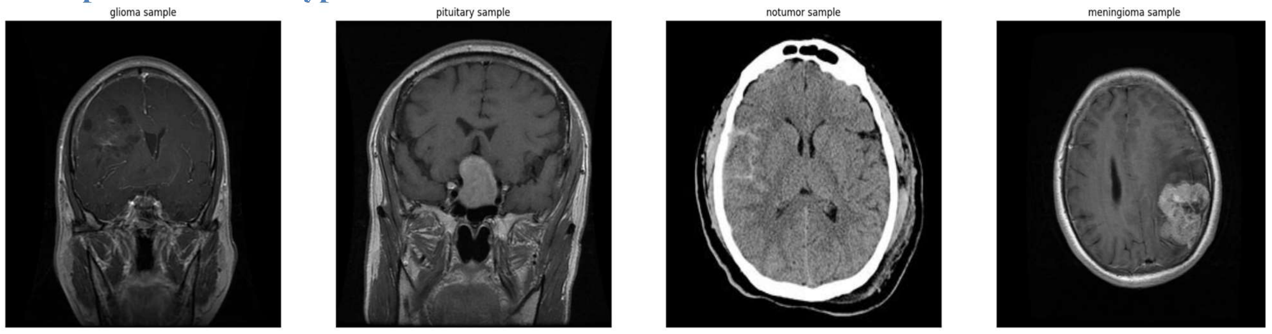
Pituitary Adenoma:

- Adenoma, a type of tumor that grows in the gland tissues, is the most common type of pituitary tumor. Pituitary adenomas develop from the pituitary gland and tend to grow at a slow rate. About 10% of primary brain tumors are diagnosed as adenomas. They can cause vision and endocrinological problems. Fortunately for patients affected by them, adenomas are benign and treatable with surgery and/or medication.

Dataset Breakdown

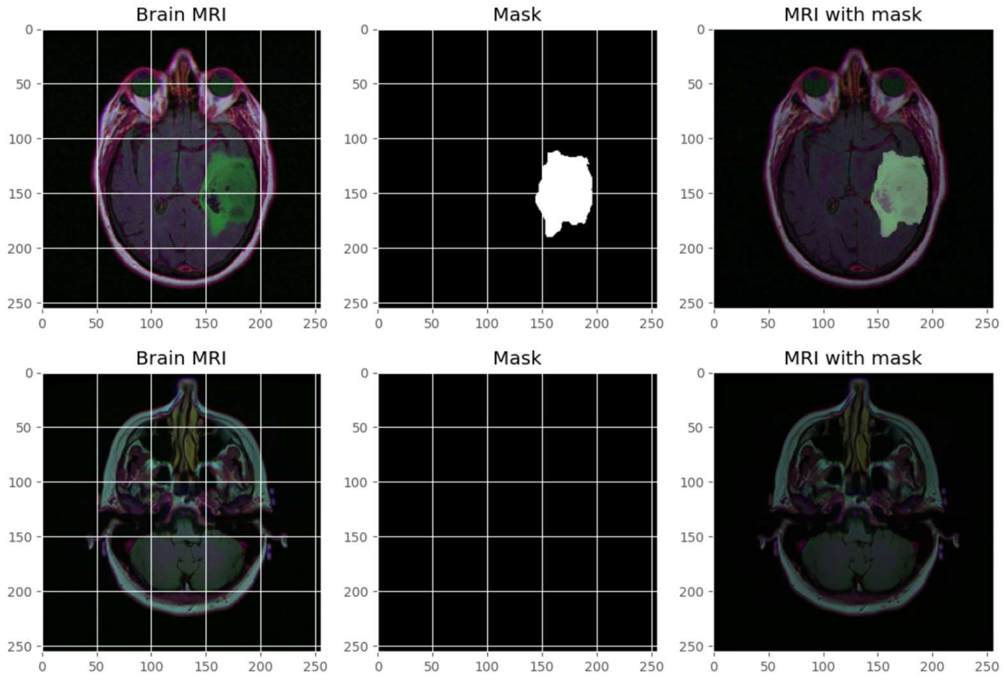
- Train Dataset:
  - 1321 files labeled as glioma
  - 1339 files labeled as meningioma
  - 1595 files labeled as no tumor
  - 1457 files labeled as pituitary
- Test Dataset:
  - 300 files labeled as glioma
  - 306 files labeled as meningioma
  - 405 files labeled as no tumor
  - 300 files labeled as pituitary

Examples of Tumor Types



Sample Image from Each Label (Glioma, Pituitary, No Tumor, and Meningioma)

Examples of Mask



MRI Images with Respective Tumor Mask

For the first set of images, there is an MRI brain scan that contains a tumor, as well as its respective tumor mask. As such the model created should be able to generate a mask of similar quality

For the second set of images, there is an MRI brain scan that does not contain a tumor, and as such there isn't an associated mask for that image. Similarly, the model should be able to determine when there isn't a tumor present in the image and as such, not generate a mask

Preprocessing

- Data Augmentation carried out by means of rotation, shifting, and reflection in order to improve model performance
- One-Hot-Encoded categorical labels generated for images
- Functions Created for Model Use:
  - conv2dBlock: Creating Custom Convolutional Layer
  - dice_coefficient: Dice Coefficient Metric to Gauge Model Performance
  - getUnet: Creating U-Net Architecture

Analysis/ Methods on Dataset 1

Custom Convolutional Layer (conv2dBlock) Structure

- 2 layers with the following characteristics:
  - Conv2D Layer
  - Same Padding
  - Batch Normalization
  - ReLU Activation

U-Net (getUnet) Structure

- Encoder Network (Contraction):
  - 5 Convolutional Layers (conv2dBlock)
  - 5 Max Pooling Layers (MaxPool2D)
  - Batch Normalization
  - Same Padding
  - ReLU Activation
  - Dropout (0.2)
- Decoder Network (Expansion):
  - 5 Transposed Convolutional Layers
  - Concatenation Layers (Reconstruct Spatial Details)
  - Filter of Stride 2
  - Batch Normalization
  - Same Padding
  - Dropout (0.2)
- Output Layer:
  - Default Convolutional Layer
  - Sigmoid Activation

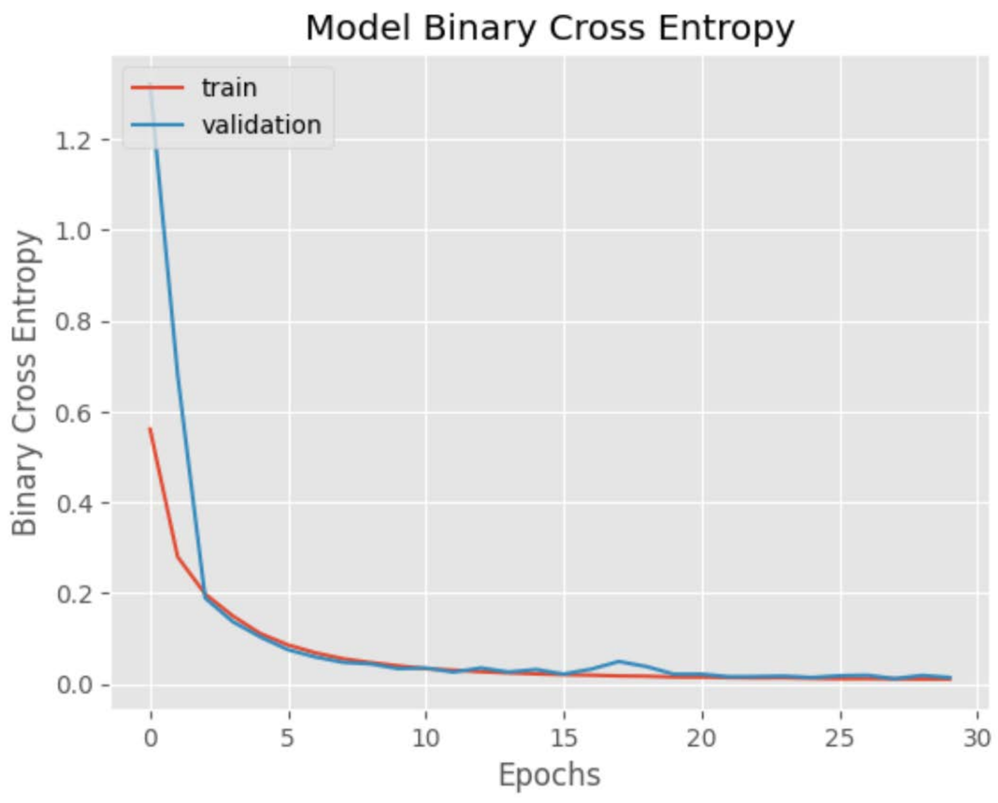
Training the Model

- The model was trained over 30 epochs
- The optimizer Adam was used with the default learning rate
- The model was trained on the following metrics: binary cross entropy, accuracy, and dice coefficient

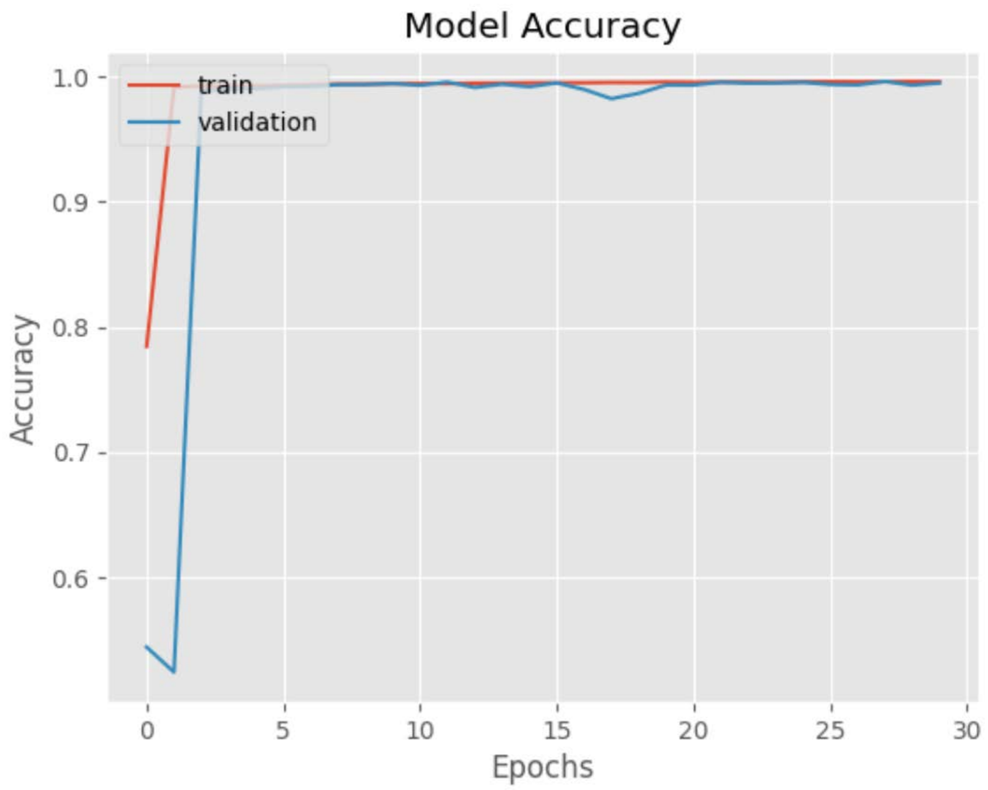
Evaluating Model Performance

- Binary Cross Entropy, Accuracy, and Dice Coefficient calculated for Train and Test data to gauge performance over 30 epochs.

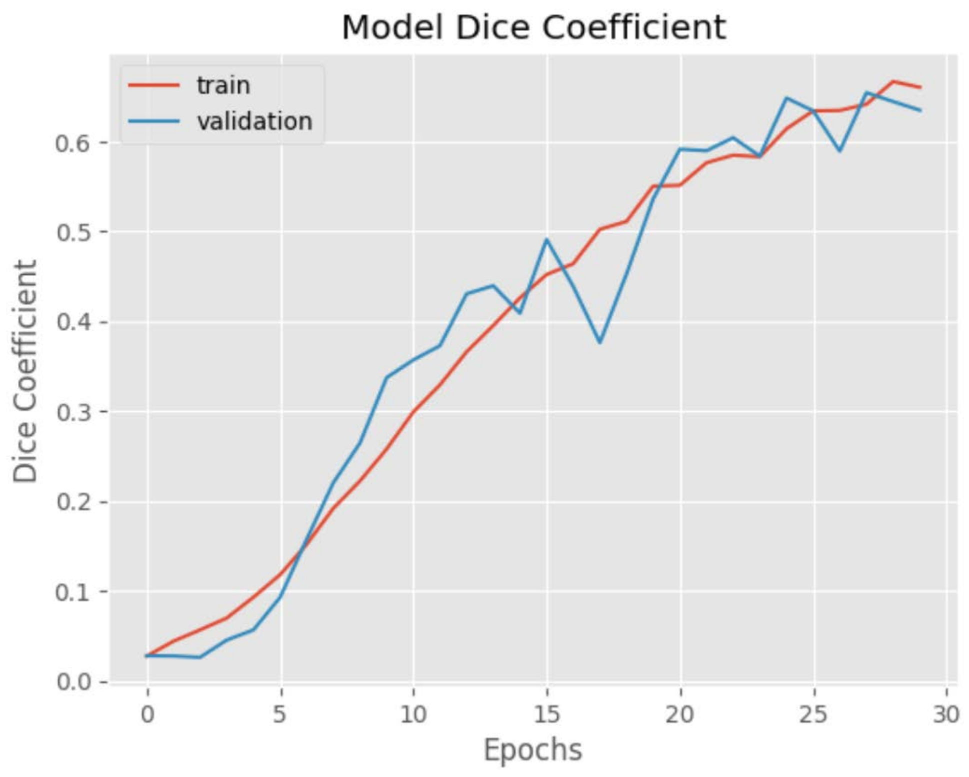
Results of U-Net Model on Dataset 1



Graph Plotting the Training and Validation Loss through each Epoch



Graph Plotting the Training and Validation Accuracy through each Epoch

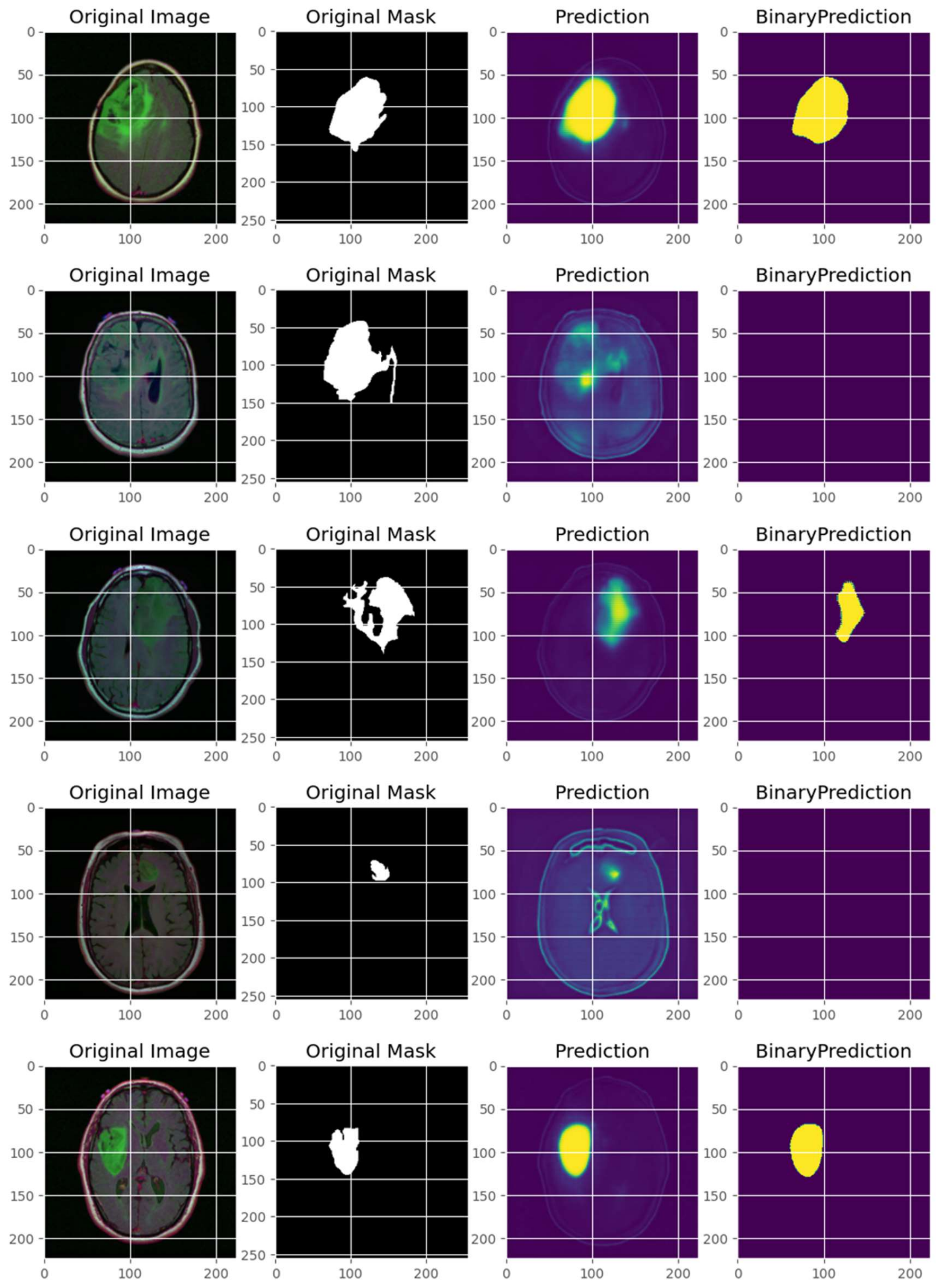


Graph Plotting the Training and Validation Dice Coefficient through each Epoch

Gauging Metrics

- Train Binary Cross Entropy: 0.0107
- Test Binary Cross Entropy: 0.0189
- Train Accuracy: 0.9961
- Test Accuracy: 0.9931
- Train Dice Coefficient: 0.6667
- Test Dice Coefficient: 0.6444

Visualizing Results



Model Predicted Outputs

Looking that the model's prediction there are some key overvations to note. The prediction model shows a probabilistic representation of the mask, where the stronger the yellow color, the more likely the pixel contains a tumor. In all of the cases above the model was able to locate potential locations of a tumor, but through the lens of the model, the probability of them actually being a tumor was low. This presents a significant issue with the model performance as is is not accurately determining a pixel to contain tumor. This is further suppoted when considering the binary prediction, which sets the threshold (in this case 0.5) for which it will classify the pixel as containing a tumor.

Ordinarily this would mean that the model would be fine tuned and trained to where it is able to accurately determine pixels to contain a tumor (with high confidence), for the purpose of this analysis it provides a baseline for comparison when applying the same model to different MRI data.

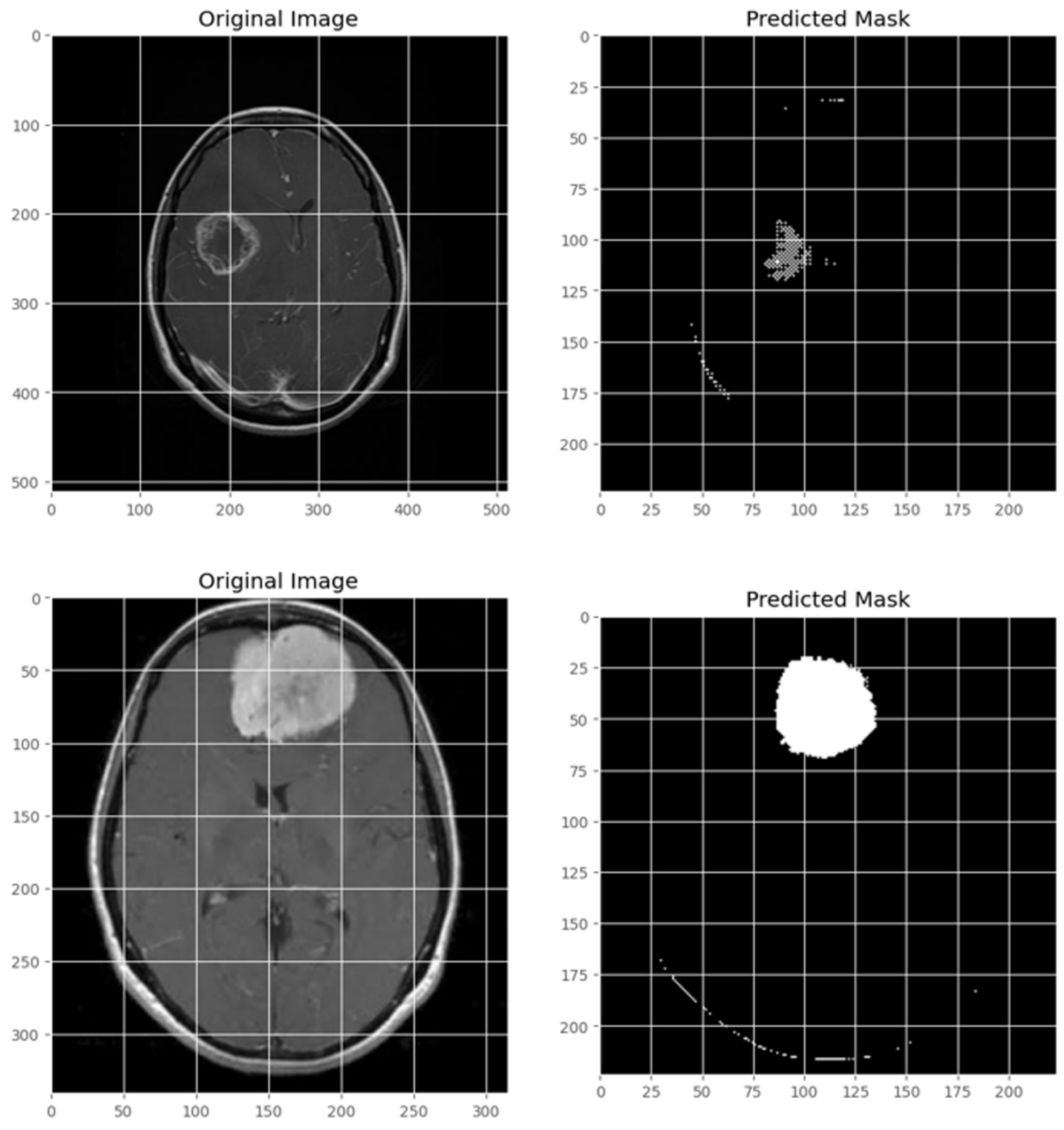
Analysis/ Methods on Dataset 2

Additional Preprocessing

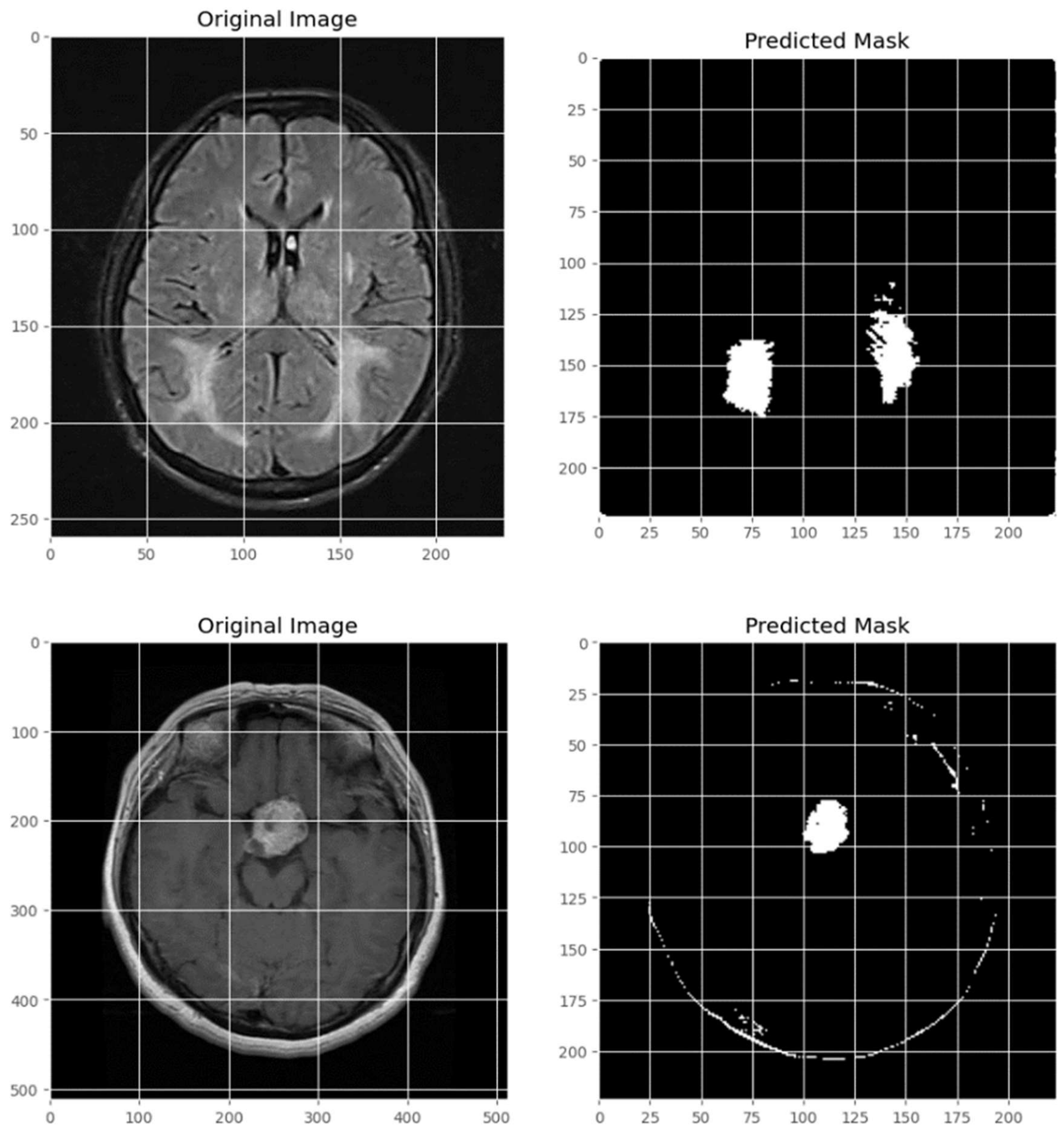
Once the U-Net model was trained, it was then applied to Dataset 2. To do this properly, the following steps were taken:

- Since the model was trained to process images with 3 channels (RGB), the grayscale images were converted to be in an RBG format (channels).
- The images were then processed to fit the model'sparameters for image height, width, and channels.
- The pathing of the images was then set to a variable and then processed through the model.

Visualizing Results



Model Predicted Outputs for Glioma (top) & Meningioma (bottom)



Model Predicted Outputs for No Tumor (top) & Pituitary (bottom)

Looking at the model’s prediction there are many similarities between this and the previous results. These images show the original and the binary prediction side by side and it is clear that the accuracy of the model falls a little more when considering these grayscale images. While the model is able to determine the general location and shape of the tumor, it is important to note that this was done after setting the threshold to 0.45, which means that the confidence has to be at least 45% or higher for it to be classified as a tumor. The threshold was set to be lower as when it was at 0.5, the images generated no masks/ very faint ones. By lowering the threshold it allows a visualization of if the model is even looking at the right places.

As such this model did not perform as well as intended and especially when considering how the model processes grayscale images, the model’s effectiveness falls of considerably.

Evaluating Process and Model

During the training process, I used the testing data to fine tune the structure of the model to create an architecture that would work best with unseen data and went through multiple iterations of adding/ removing batch normalization, dropout, and pooling layers. I also tried changing the dropout chance in certain layers to reduce over-fitting, but the effect on the performance was not particularly significant. The U-Net architecture that I ended with produced the best metrics with the testing data having a Binary Cross Entropy loss at ~0.0189, Accuracy at ~0.9931, and a Dice Coefficient of ~0.6444

Since binary cross entropy measures the similarity between predicted masks and actual masks, where values indicate closer alignment between the predicted segmentation masks and the ground truth masks, a test loss at ~0.0189 suggests a good alignment between predictions and actual masks.

Since the accuracy calculates the ratio of correctly predicted pixels to the total number of pixels in the masks, a test accuracy of ~0.9931 indicates that it correctly predicts a large portion of pixels in the masks. However, it is important to note that accuracy is not the best measure of accurately predicting the pixels as there is an inherent class imbalance with MRI images, where the number of tumor pixels compared to background pixels is low.

The Dice Coefficient is regarded as one of the better measures of segmentation accuracy, as it is sensitive to the overlap between predicted and true masks. While the other metrics (binary cross entropy and accuracy) had a very strong performance, the test dice coefficient only had a value at ~0.6444, which suggests that the predicted masks have an overlap of approximately 64.44% with the ground truth masks. This means that about 64.44% of the pixels in the predicted mask align with the pixels in the ground truth mask, which is somwhat low.

Reflection

As of now, this analysis is still incomplete, as there is much testing and fine tuning to do. During the process of developing the model I spent a lot of time fine tuning the architecture to make sure that the test data would also have a strong performance. I was hoping that if the metrics had a strong enough performance, I could apply the model to a truly unseen dataset that had similar images. While the models did perform somewhat well, they are still far from what I was expecting. In the earlier stages of analysis I didn't have the dice coefficient, and based on the loss and accuracy believed the model to be performing very well, but when visualizing the results I found that not to be the case. After implementing the dice coefficient it is clear that the model still needs work.

In the future there are a few aspects that I would want to change/ do. For the sake of runtime I only used 30 epochs, but it is important to note that the dice coefficient continued to increase and didn't seem like it was nearing its limit, so I believe that the first step would be to let the model train over more epochs and see what the limits of the metrics are. From there it would be easier to determine where to make improvements to the model. Another issue that I ran into was processing different types of images, and since the model was trained on RGB channels and applied to grayscale, I would also want to make a version of the model that is trained on the images in a grayscale channel. While it would take some time, I think that continuing to play around with the architecture and processing the data in different ways would result in a far better segmentation/ mask generation.