Springboard

Data Science Cohort 2024 – 2025

Capstone Project III

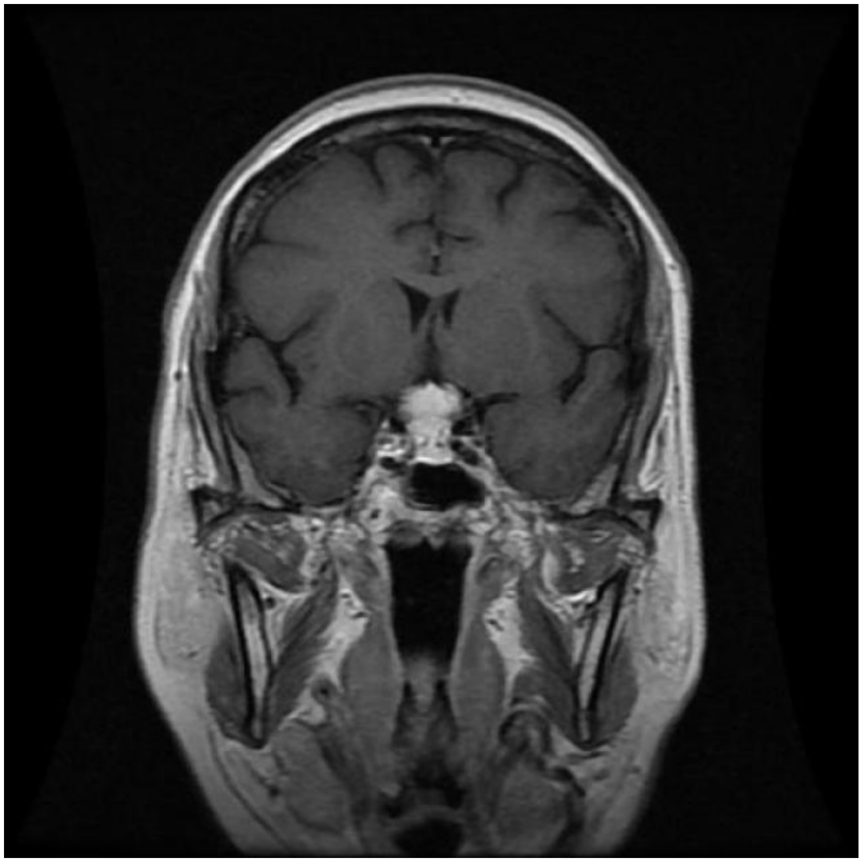
Ofri Oren

Brain Tumor Image Segmentation

Summary:

In recent years, cancer rates have increased among younger people under sixty. Although no one is sure what is causing these changes, it is a significant change. I found a pre-labeled dataset of MRI scans on Kaggle with segmentation of brain tumors and classification of the type of tumor based on location. Instead of looking at classification, I began with the more challenging question of how to segment images. I used machine learning with image specific modules like TensorFlow, Keras, Albumentations, and did substantial post-processing but was unable to train a machine learning to identify and create an accurate segmentation mask for tumors. I will next attempt to use anomaly detection to attempt to isolate the tumor.

Outline:



Pituitary tumor from training dataset

[https://www.kaggle.com/datasets/bilalakgz/brain-tumor-mri-dataset/data] accessed on 2 Febrary 2025.

Background

Recent news from almost every major medical journal describes cancer rates among younger people rising.[1] However, there is little information as to what caused this change in 'early-onset' cancer. Even though brain cancer is not one of the most common types of cancer

[1] Katella (August 2024). "What to know about the rising rates of early-onset cancer." [https://www.yalemedicine.org/news/early-onset-cancer-in-younger-people-on-the-rise]; Bartosh (August 2024). "Why are more young people getting cancer? What to know as cases rise." [https://www.uchicagomedicine.org/forefront/cancer-articles/why-are-more-young-people-getting-cancer]; Cox (October 2024). "'These are people in the prime of life': The worrying puzzle behind the rise in early-onset cancer." [https://www.bbc.com/future/article/20241004-the-puzzle-of-rising-early-onset-breast-and-colorectal-cancer-in-younger-people]; amongst several others.

amongst young people, this piqued my interest. I used a pre-labelled dataset from Kaggle to begin examining how to identify tumors in brain scans from MRI imaging.

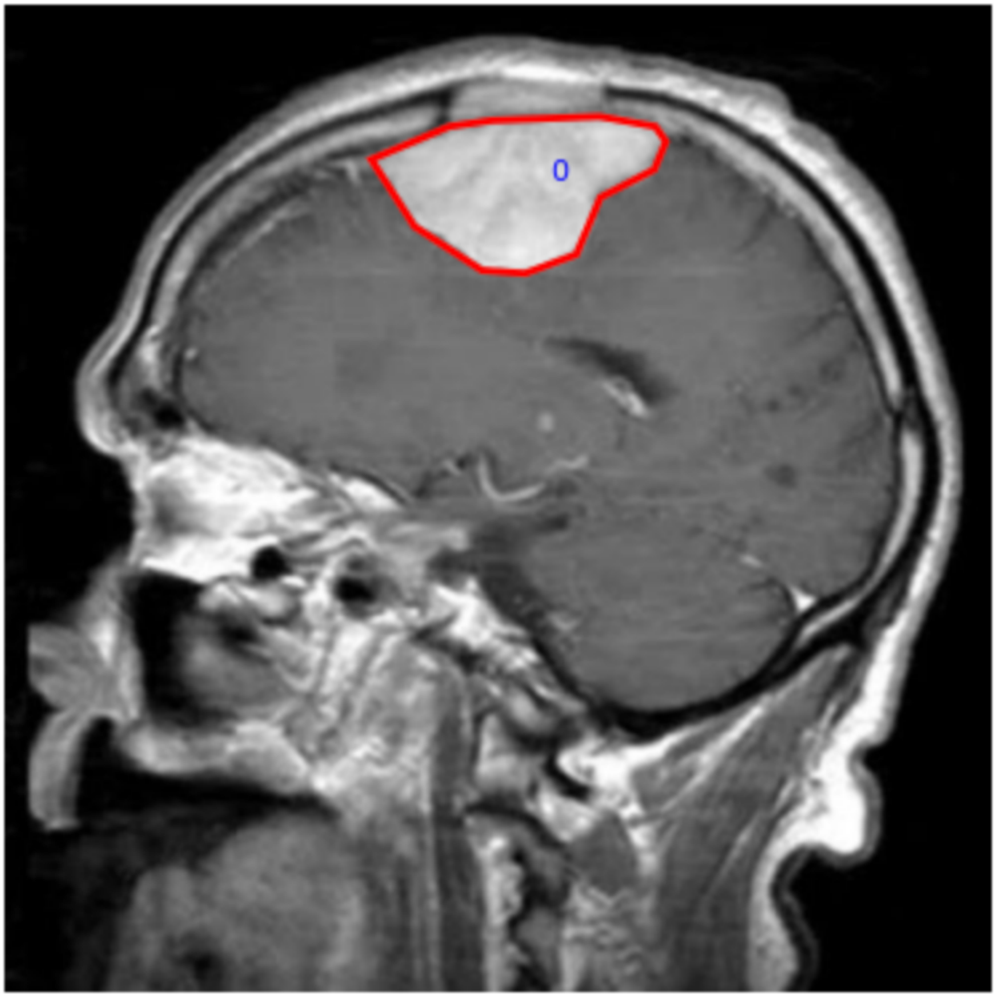
Data

I used three different datasets from Kaggle. In the first dataset, the tumors are identified and the coordinates of the tumor are listed in a text file that corresponds to the image.[2] In the second dataset there are a couple of thousand unlabeled images of brains with tumors.[3] In the last dataset I tried there are images of brains from MRI scans, which do *not* have tumors or any other abnormalities.[4] All of the files have jpegs of the images, while the second dataset also has a comma-separated-value file with information about the tumor, I only used the jpeg images.

Exploratory Data Analysis

I inspected the image files from the training dataset and used their labels to create polygons around the tumor. In the image segmentation subset of the data all tumor files had appropriate labels. There were images for three types of tumors including pituitary, glioma, meningioma, and images of brains without tumors. There were 827 images of pituitary tumors, 826 glioma, 822 meningioma, and 395 no-tumor images. There does seem to be a class imbalance here between tumor and non-tumor images, but it favors the classes that need image segmentation and should thus help the machine-learning system to identify tumors in a regular segmentation model. In the image below you can see a brain image with the tumor identified with the pre-labeled segmentation.

[2] https://www.kaggle.com/datasets/bilalakgz/brain-tumor-mri-dataset/data
[3] https://www.kaggle.com/datasets/jakeshbohaju/brain-tumor?select=Brain+Tumor.csv
[4] https://www.kaggle.com/datasets/trainingdatapro/dicom-brain-dataset



Processing and Modelling

The processing and modelling of a U-net neural net system was quite complicated and not ultimately successful in segmenting MRI images. I started with the images at size, 640x640 pixels, and ran them through a neural net with the following architecture:

Model: "functional"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 640, 640, 3)	0	-

conv2d (Conv2D)	(None, 640, 640, 32)	896	input_layer[0][0]
conv2d_1 (Conv2D)	(None, 640, 640, 32)	9,248	conv2d[0][0]
max_pooling2d (MaxPooling2D)	(None, 320, 320, 32)	0	conv2d_1[0][0]
conv2d_2 (Conv2D)	(None, 320, 320, 64)	18,496	max_pooling2d[0]…
conv2d_3 (Conv2D)	(None, 320, 320, 64)	36,928	conv2d_2[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 160, 160, 64)	0	conv2d_3[0][0]
conv2d_4 (Conv2D)	(None, 160, 160, 128)	73,856	max_pooling2d_1[…
conv2d_5 (Conv2D)	(None, 160, 160, 128)	147,584	conv2d_4[0][0]
conv2d_transpose (Conv2DTranspose)	(None, 320, 320, 64)	32,832	conv2d_5[0][0]
concatenate (Concatenate)	(None, 320, 320, 128)	0	conv2d_transpose… conv2d_3[0][0]
conv2d_6 (Conv2D)	(None, 320, 320, 64)	73,792	concatenate[0][0]
conv2d_7 (Conv2D)	(None, 320, 320, 64)	36,928	conv2d_6[0][0]

conv2d_transpose_1 (Conv2DTranspose)	(None, 640, 640, 32)	8,224	conv2d_7[0][0]
concatenate_1 (Concatenate)	(None, 640, 640, 64)	0	conv2d_transpose… conv2d_1[0][0]
conv2d_8 (Conv2D)	(None, 640, 640, 32)	18,464	concatenate_1[0]…
conv2d_9 (Conv2D)	(None, 640, 640, 32)	9,248	conv2d_8[0][0]
conv2d_10 (Conv2D)	(None, 640, 640, 1)	33	conv2d_9[0][0]

```
Total params: 466,529 (1.78 MB)
Trainable params: 466,529 (1.78 MB)
Non-trainable params: 0 (0.00 B)
```

I saved the model checkpoints and found the correct number of epochs based on training and validation loss was one. Unfortunately, this model was not able to segment the image, my pixel-wise accuracy was zero percent, my Jacquard-index measure of 'IoU', and Dice coefficient were also zero. I tried to adjust my performance results in post-processing by looking at the histogram of pixel-coloration distribution and adjusting the threshold but was unable to see any changes even after calculating the optimal threshold value of 0.45 instead of the standard 0.5 value.

Next, I tried to increase the quantity of images for training by transforming the images by rotation, flipping, and stretching images. I resized the images down to 256x256 for faster processing. Because my images are in black and white I used a sigmoid activation on my final neural layer, with binary cross-entropy loss, and only one class for the classification. I was unable to see any changes in output. I adjusted and tried different loss metrics, I tried different types of post-processing, and different values for thresholding at different points in the post-processing; while I was able to see changes from an all-black or all-white output segmentation mask, I was not able to significantly alter the Jacquard index scores. I attempted to overfit on a single image and small subset of the data, but again nothing changed the model performance substantially. I also tried to use the second dataset of unlabeled data to train the model and create pseudo-masks. However, there was not a sufficient amount of pseudo-masks accurate enough to warrant training the model on the second dataset. I found the problem might be more complicated than machine-learning can presently handle with this approach.

Suggestions

I suggest, and will attempt, solving this problem by using anomaly detection instead of trying to train the machine to segment the image. I recommend creating color distinctions for tumors instead of a polygonal outline with coordinates. I also suggest using MRI scans with staining instead of just black and white images. In black and white images other features like the spinal column might be mistaken for tumorous white blobs, however with staining grey-matter is distinct from bones.