MRI Brain Tumor Classification

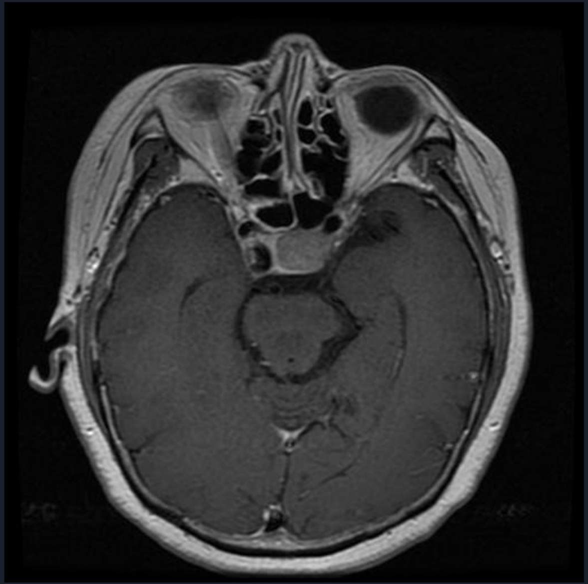
Karthik Raj

Significance

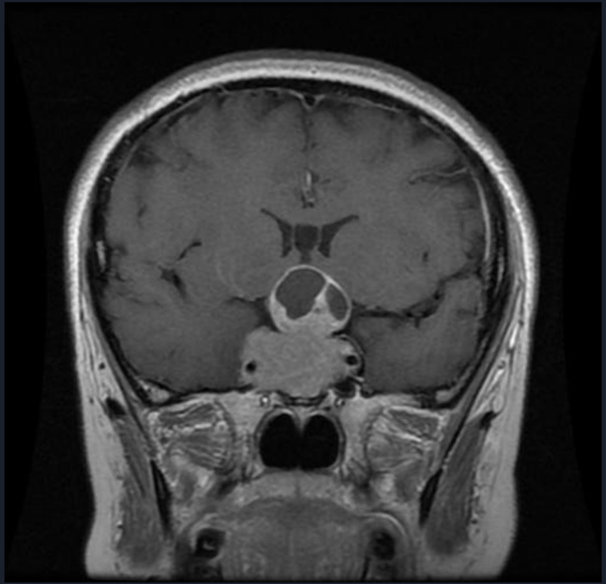
- Based on 255 uninterrupted eight-hour workdays per year, radiologists are needing to review one image every three to four seconds to meet workload demands. (2015)
  - Overall workload per radiologist is increasing as
    - MRI and CT scans are becoming more prevalent (2021)
    - Increase in complexity of information gleaned from scans (2021)
- Increasing cases of burnout amongst diagnostic radiologists (2021)
- Potential for deep learning to mitigate these issues

General Characteristics of Dataset

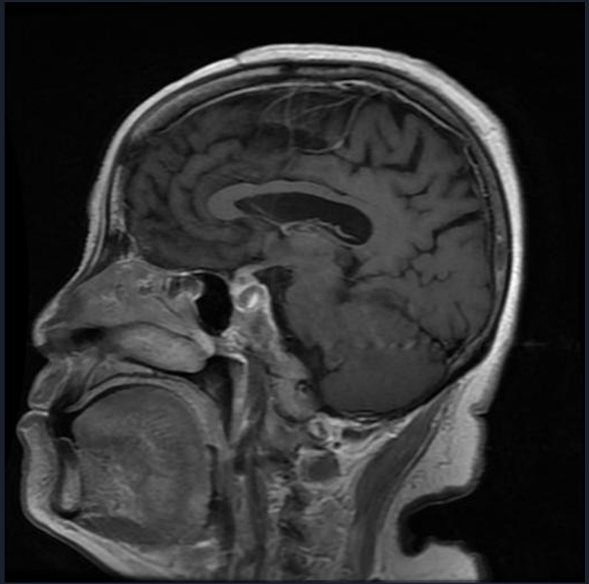
- Tumor associated classes are mostly 512 by 512
- Non-tumor class has randomly sized images
- Scans are taken at different orientations



Axial



Coronal



Sagittal

Dataset Breakdown

Training: 5712 Images -> **Split into ⅔ Training & ⅓ Validation**

Glioma	Meningioma	Pituitary	No Tumor
1321	1339	1457	1595

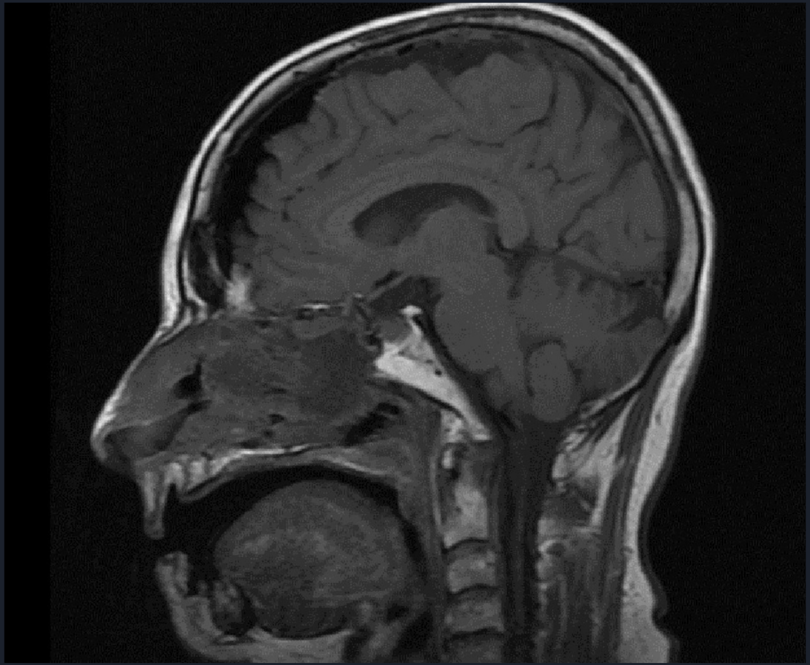
Testing: 1311 Images

Glioma	Meningioma	Pituitary	No Tumor
300	306	300	405

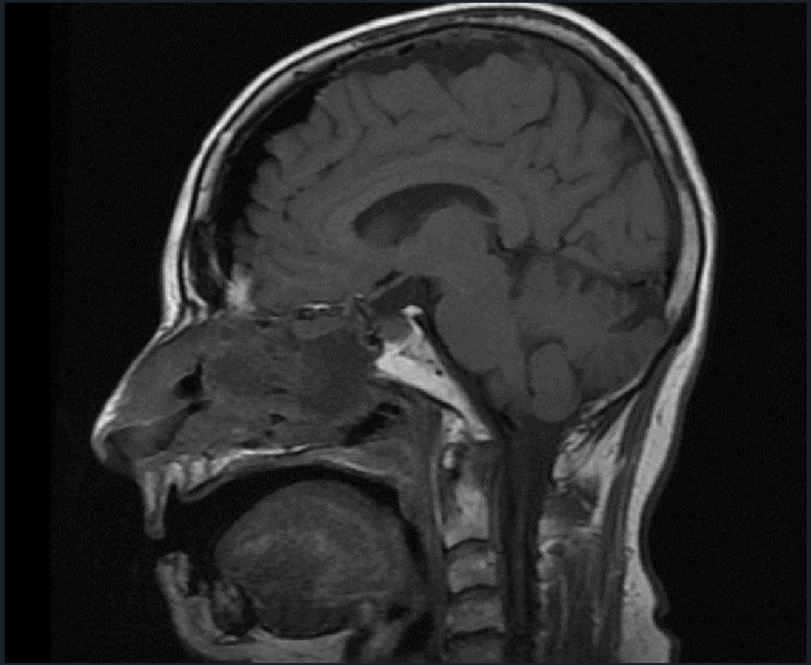
Goal: Accurately classify an MRI Brain image into one of these 4 classes

- **Tumor classes vary based on location in the brain**

Data Processing Walkthrough

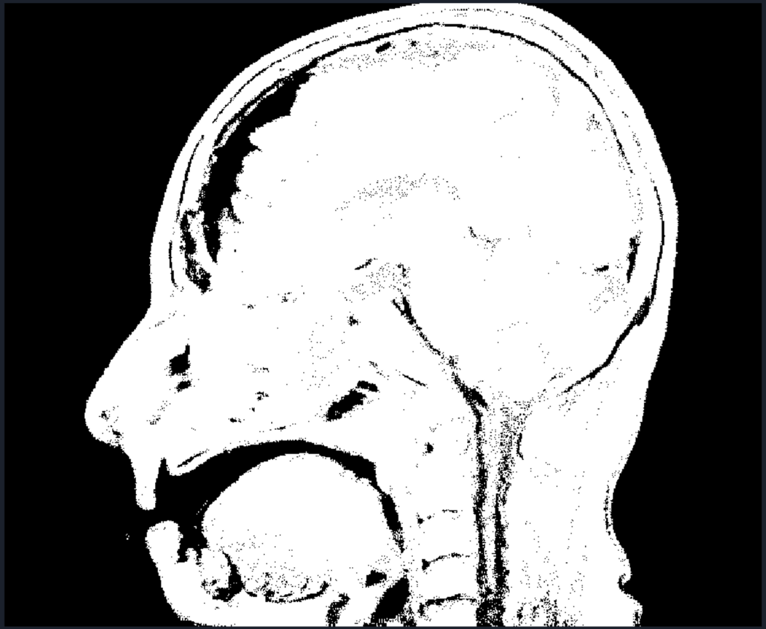


Original Image: (741, 900, 3)

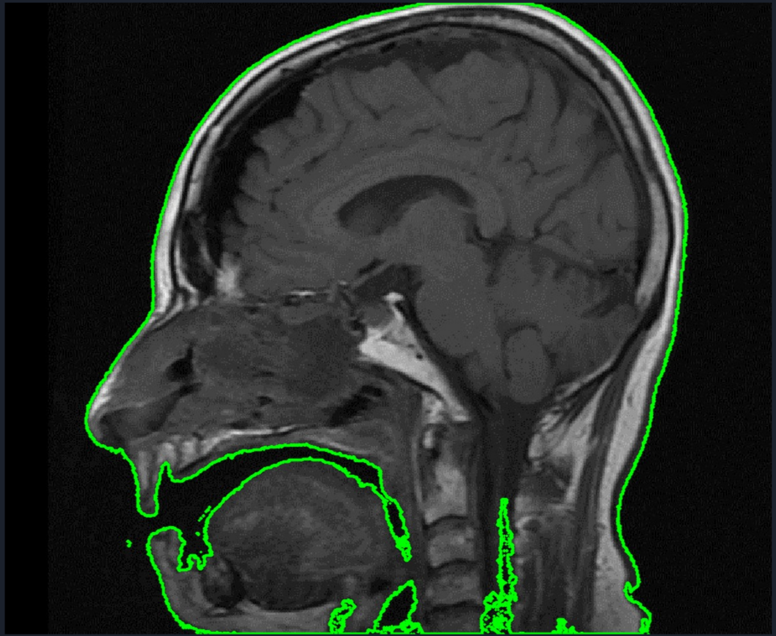


Grayscale Image: (741, 900)

Data Processing Walkthrough

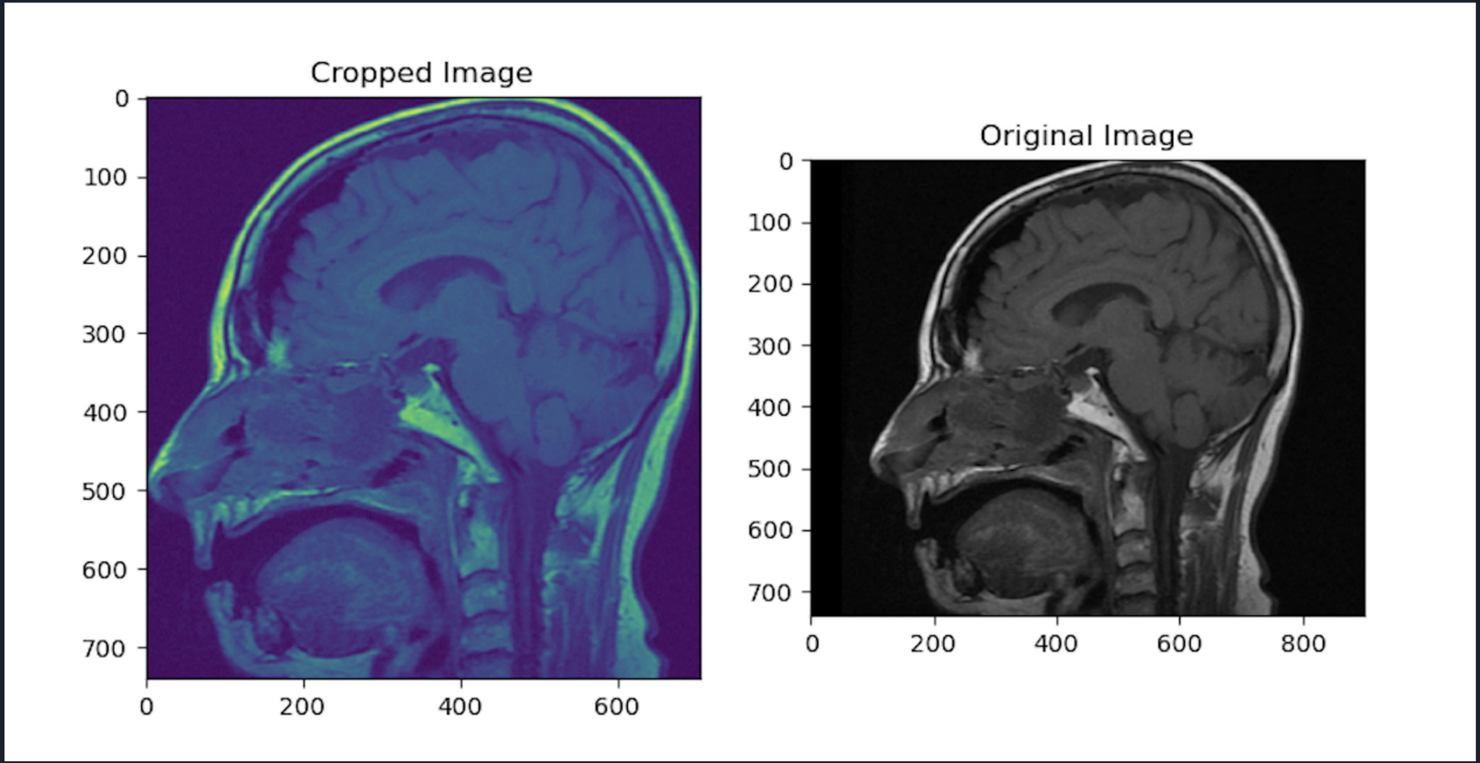


Apply Binary Mask:
If pixel intensity < 30, set pixel to 0 (black)
Else: set pixel to 255 (white)



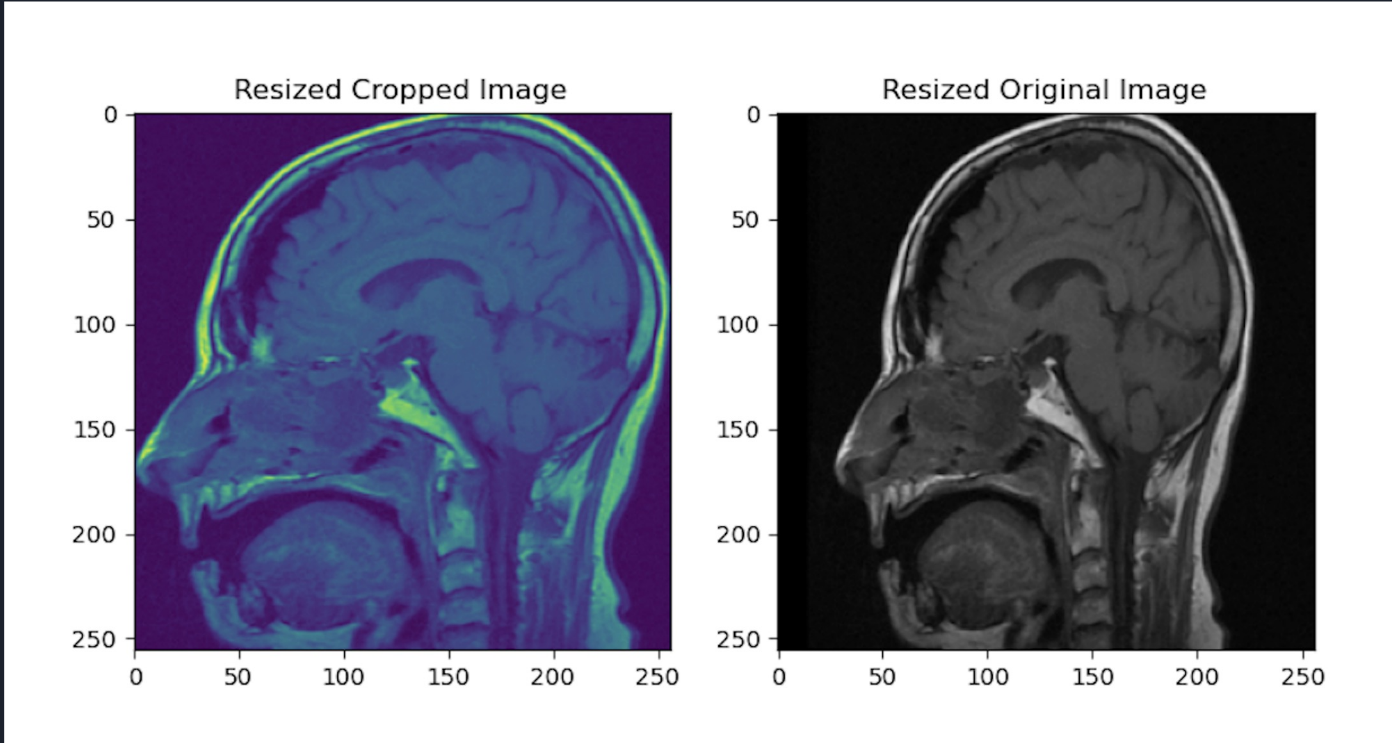
Draw largest contour on ROI

Data Processing Walkthrough



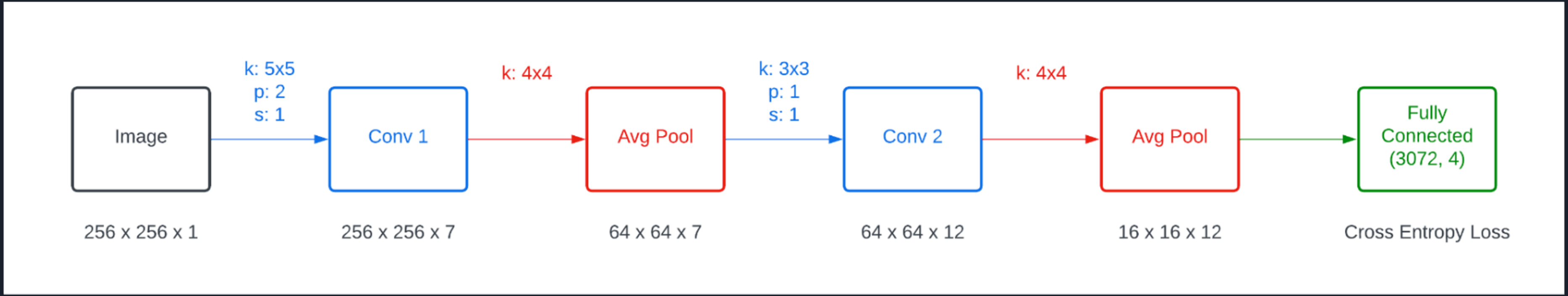
Cropping to ROI

Data Processing Walkthrough



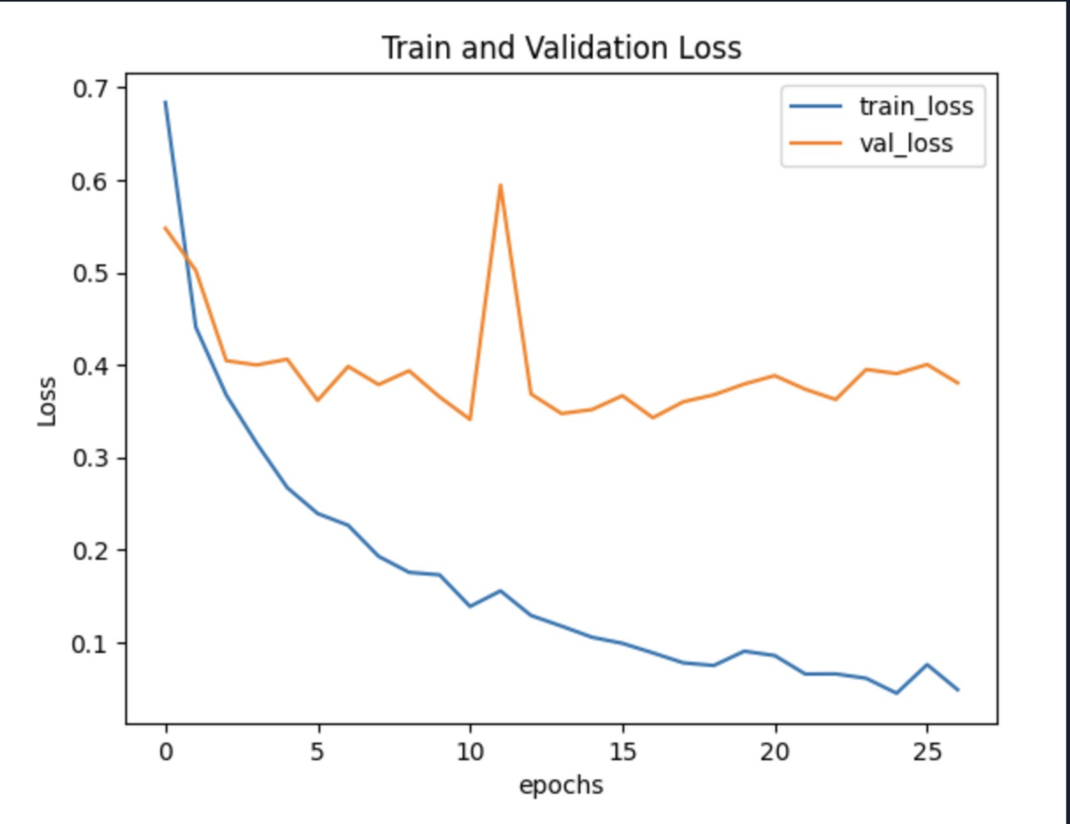
Cropping images to (256,256) & Removed Extraneous Image Background

Convolutional Neural Network



Used Batch Normalization & ReLU as the Activation Function throughout Convolution Layers

(⅔) Training & (⅓) Validation

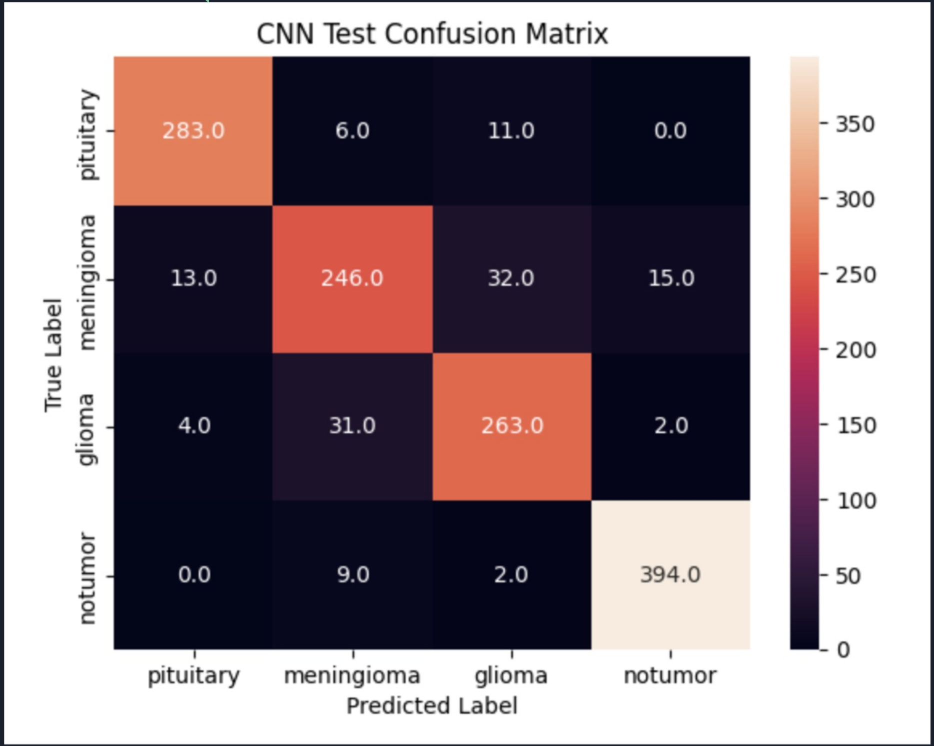


My Other Versions of CNN

	epochs	batch_size	LR	avg_train_acc	avg_val_acc	L2	num_layers	num_filters	kernel_sizes
19	27	16	0.001	0.984156	0.918943	True	2	7 & 12	5 & 3
21	27	16	0.001	0.981618	0.915616	True	2	7 & 12	5 & 3
17	27	16	0.001	0.984769	0.914741	True	2	7 & 12	5 & 3
20	27	16	0.001	0.984331	0.914566	True	2	7 & 12	5 & 3
18	27	16	0.001	0.985294	0.913691	True	2	7 & 12	5 & 3
0	28	16	0.001	0.983894	0.911590	True	2	7 & 12	5 & 3
5	28	16	0.001	0.982230	0.911415	True	2	5 & 10	9 & 5
16	27	16	0.001	0.982668	0.910889	True	2	7 & 12	5 & 3
8	24	16	0.001	0.991246	0.910889	True	2	10 & 15	9 & 5
7	24	16	0.001	0.984681	0.909839	True	2	7 & 12	9 & 5

Classification Metrics on Test Dataset

Accuracy: 90.5%



	precision	recall	f1-score	support
pituitary	0.94	0.94	0.94	300
meningioma	0.84	0.80	0.82	306
glioma	0.85	0.88	0.87	300
notumor	0.96	0.97	0.97	405
accuracy			0.90	1311
macro avg	0.90	0.90	0.90	1311
weighted avg	0.90	0.90	0.90	1311

Potential Next Steps

1. Leverage PyTorch Dataset Class & DataLoaders to increase code efficiency and expedite overall training process
2. Try a transformer-based approach and see how a more advanced architecture compares to current model generated from scratch
3. Flagging potentially misclassified images:
   a. Characterize a model unsure state as having largest 2 softmax probabilities within 10% of each other
   b. Show radiologist the array of these images for his/her diagnosis

How It would Look like?

```
In [107]: def unsure_image(model_output):
              probs = nn.Softmax(dim = 1)(model_output).cpu().numpy()
              largest_2_probs = np.sort(probs)[:, -2:]
              unsure_ind = np.squeeze(np.diff(largest_2_probs) < 0.10).nonzero()[0]
              return unsure_ind

In [108]: uns_inds = unsure_image(model_output= model_outputs)
          uns_inds

Out[108]: array([ 89, 159, 316, 347, 373, 461, 541, 567, 578, 588, 647, 674, 692,
                 715, 838])

In [109]: pred_mod_uns = torch.argmax(nn.Softmax(dim = 1)(model_outputs), dim = -1).cpu().numpy()[uns_inds]
          pred_mod_uns

Out[109]: array([1, 2, 1, 1, 1, 2, 3, 2, 2, 2, 1, 2, 1, 2, 1])

In [110]: ans_mod_uns = (test_y[uns_inds]).astype(int)
          ans_mod_uns

Out[110]: array([0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2])

In [111]: decimal_mod_uns_correct = np.sum(pred_mod_uns == ans_mod_uns) / len(ans_mod_uns)
          decimal_mod_uns_correct

Out[111]: 0.3333333333333333
```

References

1. McDonald RJ, Schwartz KM, Eckel LJ, et al. The Effects of Changes in Utilization and Technological Advancements of Cross-Sectional Imaging on Radiologist Workload. *Acad Radiol.* Published online July 22, 2015.
2. Kwee, T.C., Kwee, R.M. Workload of diagnostic radiologists in the foreseeable future based on recent scientific advances: growth expectations and role of artificial intelligence. *Insights Imaging* 12, 88 (2021). https://doi.org/10.1186/s13244-021-01031-4

Questions???