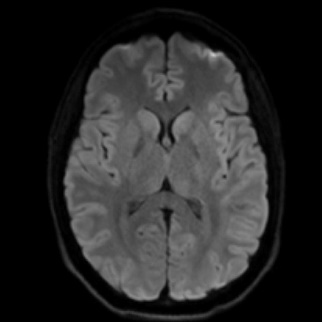
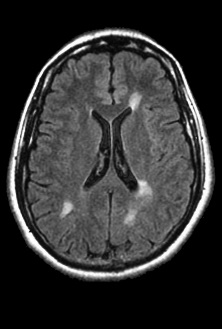
Image Processing Project

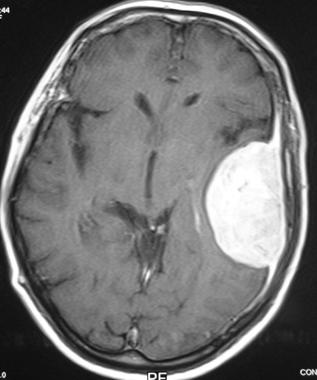
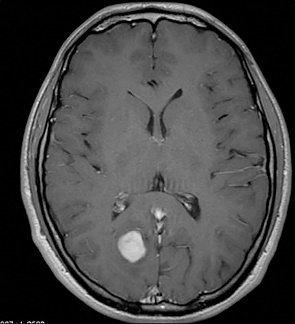
Shido Nakajima

05/01/2023

05/01/2023

**ABSTRACT:** Using a dataset of 253 brain MRI images with and without tumor, build and train a neural network for detecting whether an input brain MRI image has tumor or not. Below are the five images of the classified images (3 without tumor, 2 with tumor).

A close-up of the moon

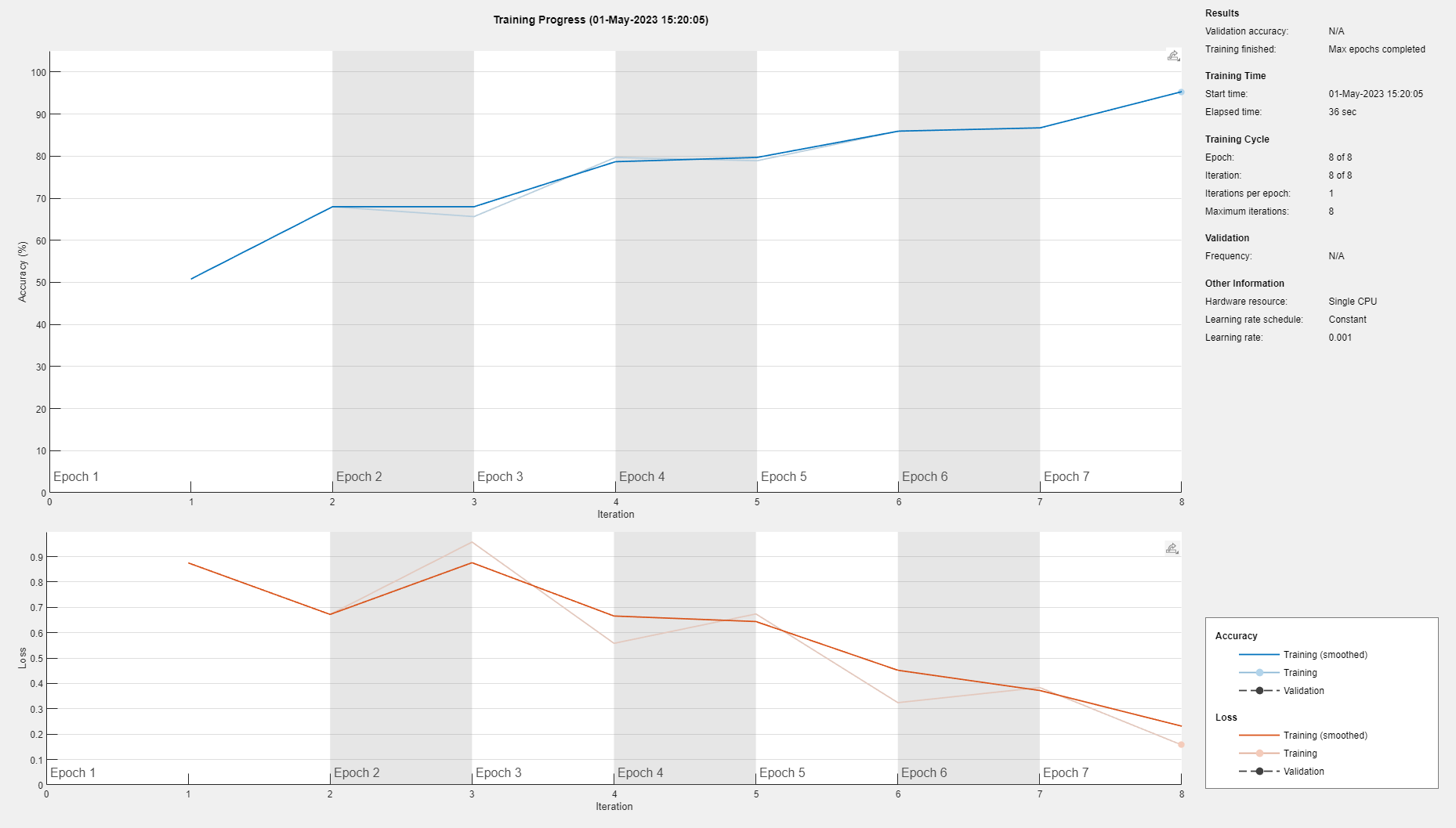
Description automatically generated with low confidence

**METHODS:** Using lab 11, “Training a Model from Scratch” MathWorks page as a reference, I started with the implementation of the neural network. The MRI image database was downloaded from Kaggle page “Brain MRI Images for Brain Tumor Detection”, which had a dataset with folders named “yes” and “no” (renamed “tumor” and “no tumor” for convenience).

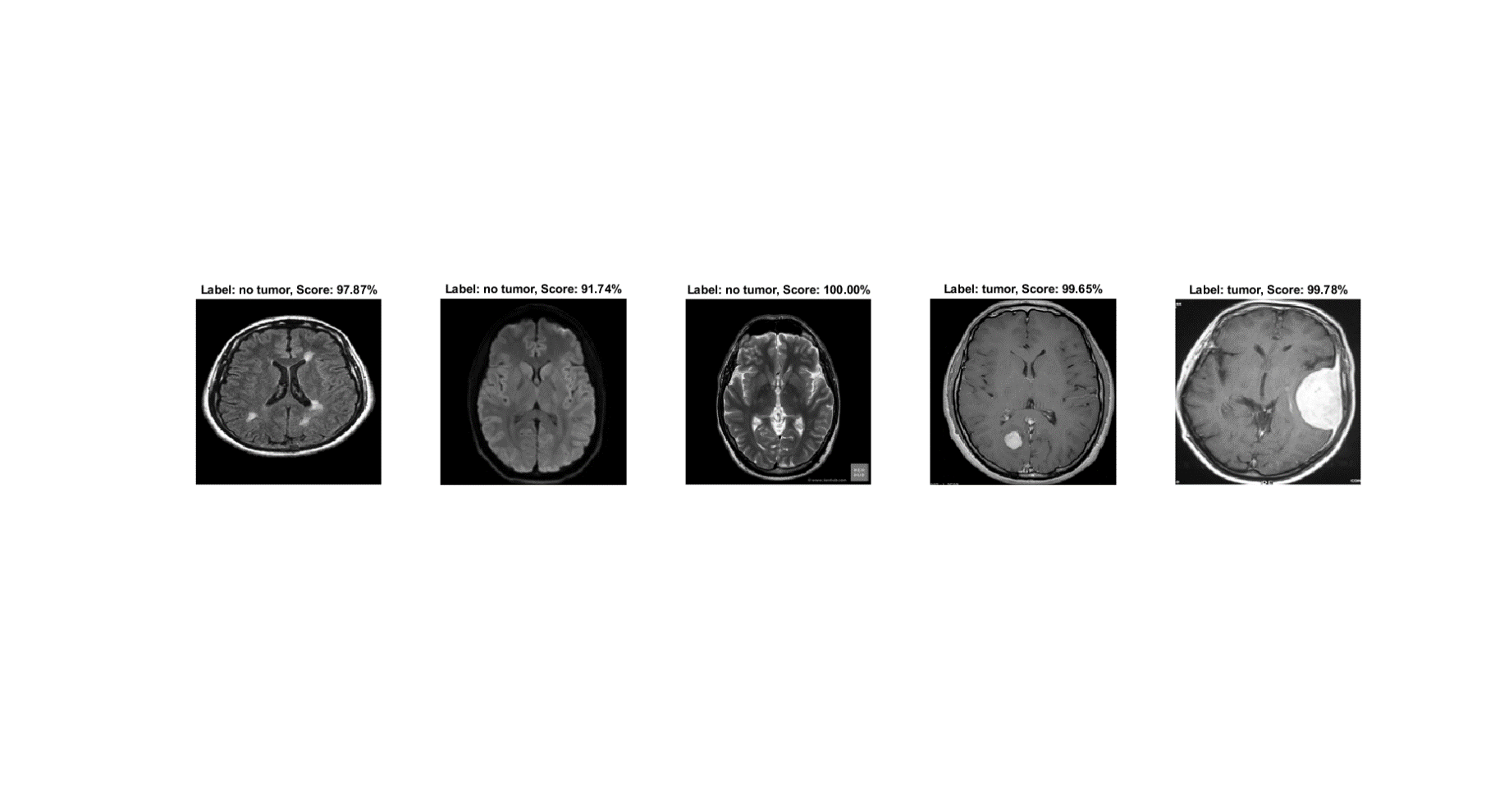
The downloaded dataset included images with varying size and color scale, and thus I was unable to use the code from MathWorks page directly. Instead, I used augmentedImageDatastore() function with targetSize being [200, 200] for the image size and the option ‘rgb2gray’ to create a new image datastore with same database images but resized to 200x200 pixel grayscale images. Initially I tried using two transform() functions with parameters imresize() and rgb2gray(), but the transform() function merely output datastores of the transform functions rather than image datastores containing the transformed images.

While the augmentedImageDatastore() function was able to output a transformed image datastore, the training was unable to run due to an error with the data being split using splitEachLabel. However, the training ran with no error once this section of code was commented out. Since the datastore was not split into training data and validation data, the training plot seems very simple as compared to the training plot yielded from lab 11 but is able to classify input image with accuracy output.

For the five brain MRI images acquired from web, I simply used imresize() and rgb2gray() functions to make 5 different variables rather than making a datastore, as making another datastore would be more unnecessary work. Since one of the images acquired was already in gray scale, the rgb2gray() function was used 4 times rather than 5 times. Once the classification was conducted on each image using classify() function, the Ypred and score variables were used to acquire the classification label and score. On a single figure with 5 subplots, the five resized and grayed images were displayed, with titles displaying the classification label and score using sprint() function to allow use of Ypred variable as formatted string.

**RESULTS:** 

Above is the training plot yielded. It does not have a validation accuracy because the image datastore was not split into training and validation datastore.



Above is the 5 images with corresponding output images. From the left, the titles are: “Label: no tumor, Score: 97.87%”; “Label: no tumor, Score: 91.74%”; “Label: no tumor, Score: 100.00%”; “Label: tumor, Score: 99.65%”; “Label: tumor, Score: 99.78%”. The image in the center has a score of 100%, which may indicate that this image is in the dataset. However, I have run this code a few times and each yielded a different score due to the random nature of the neural network training, with the score for the image in the center going as low as 97%.

**CONCLUSION:** The project was successful: I was able to create and train a neural network to classify if a given brain MRI image has tumor or not. I learned that the variance in the neural network training can cause the score to deviate, although the deviation is not very large. These techniques would be useful in clinical applications in the field of image diagnosis. By feeding more images into the image dataset, one would be able to increase the amount of training images and thus increase the accuracy of the detection. This would allow even a non-expertise in brain tumor diagnosis to make accurate diagnosis of brain tumor.

**APPENDIX:** During the Mid-project check, I learned that I would have to resize the images in the dataset. While looking through the Kaggle database together with a classmate, we noticed that the image resolution was varying quite a lot and remembered that the neural network only accepts a certain image size.

**REFERENCES:**

Chakrabarty, N. (2019, April 14). *Brain MRI images for Brain tumor detection*. Kaggle. Retrieved May 1, 2023, from <https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection?resource=download>

Dominguez, M. (n.d.). *Meningioma*. Meningioma - Oncology - Medbullets Step 2/3. Retrieved May 1, 2023, from <https://step2.medbullets.com/oncology/120395/meningioma?hideLeftMenu=true>

Selvapandian, A., & Manivannan, K. (2018). Performance analysis of meningioma brain tumor classifications based on gradient boosting classifier. *International Journal of Imaging Systems and Technology*, *28*(4), 295–301. <https://doi.org/10.1002/ima.22288>

Vaskovic, J. (2023, April 12). *Normal brain mri*. Kenhub. Retrieved May 1, 2023, from <https://www.kenhub.com/en/library/anatomy/normal-brain-mri>

WETA Public Television. (2021, October 26). *Brain imaging: What are the different types?* BrainLine. Retrieved May 1, 2023, from <https://www.brainline.org/slideshow/brain-imaging-what-are-different-types>

Wexler, M. (2023, April 24). *MRI and MS diagnosis: Does MS show up on MRI?: Multiple...* Multiple Sclerosis News Today. Retrieved May 1, 2023, from <https://multiplesclerosisnewstoday.com/multiple-sclerosis-diagnosis/mri-magnetic-resonance-imaging/>