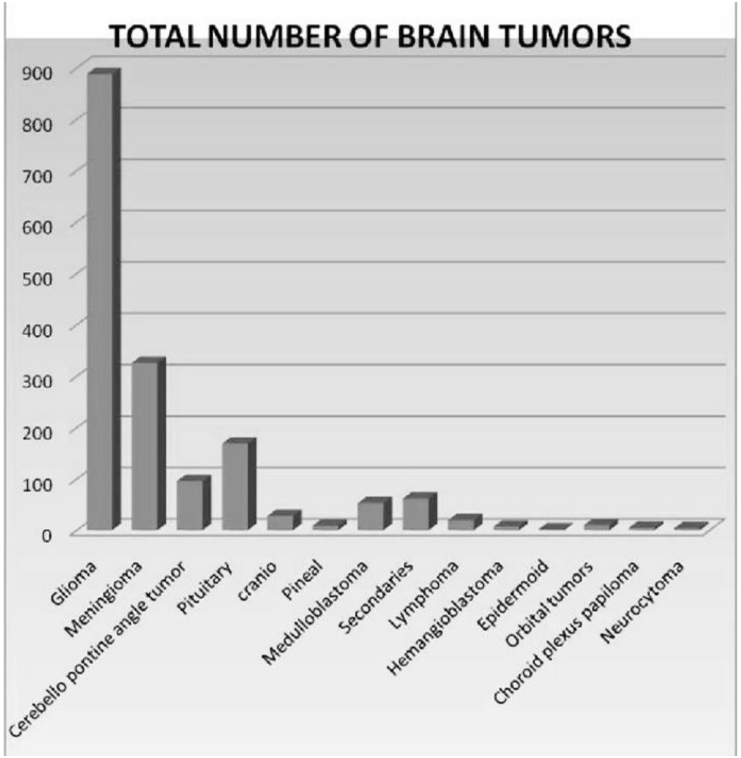
Brain Tumour Detection

Description of problem

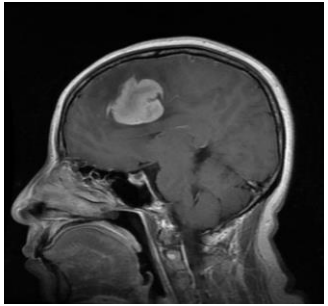
- Brain tumors account for over 85% of all Central Nervous System tumors. With the 5 year survival rate at 34% for men and 36% for women.
- The best detection method for brain tumors is using Magnetic Resonance Imaging (MRI). As useful as these scans are, when manually read, they can be prone to errors due to the complexities in brain tumors.
- To help automate this process and increase accuracy, we developed a machine learning model. Which when given an MRI scan can not only accurately diagnose the patient to an accuracy of 85%, but also which type of tumor it is.
- We constructed our model using a CNN model as The accuracy of it was relativity higher compared to the normal neutral network.



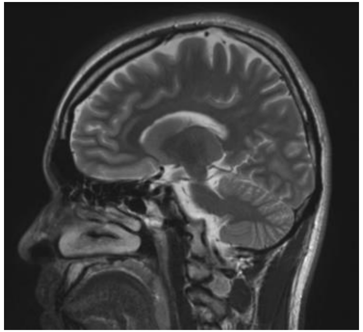
Description of data

- The dataset we choose is composed of over 3000 images of varying brain tumor diagnoses. The dataset was divided into training, test and validation folders, and under each were the 4 categories(no_tumor, pituitary_tumor, meningioma_tumor, glioma_tumor).
- We combined the training and testing folders and divided it ourselves to ensure new training, test, and validation sets each time we ran the model.
- The images are generated MRI scans that are examined by radiologist. We read those images and sent them into the model as a stream of pixels.

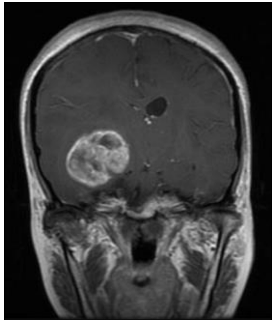
Dataset taken from: *https://www.kaggle.com/sartajbhuvaji/brain-tumor-classification-mri*



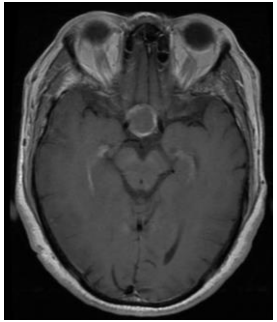
Meningioma Tu



No Tumor

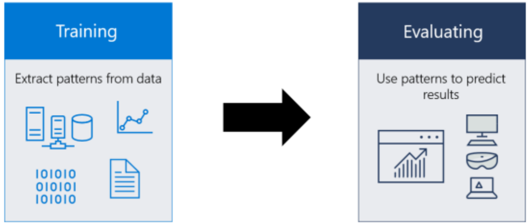


Glioma Tumor



Pituitary Tumor

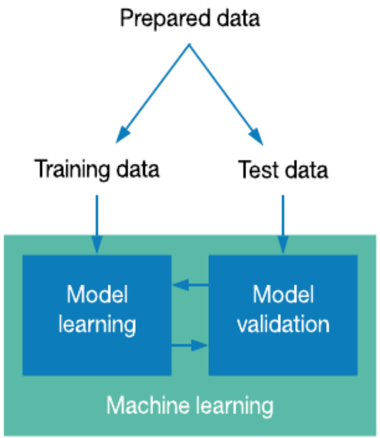
Description of design



- For the model, we decided to go with a convoluted neural network with the image size set to (200, 200), and we also set the input layer to (200, 200, 3) to accommodate for the RGB values for each pixel.
- We used a total of 4 convolution2d layers to the detect as many features in our input.
- Each layer had subsequently greater filter than previous one and this is so it could pick up on individual nodes accurately.
- The strides among those layers were set to 1 as we wanted keep as much information as we could from the image
- Padding was also set to “Same” for all the layers, since we wanted to pass the same dimension through other convolution layers
- We also used multiple MaxPooling2d layers with a pool size of (2,2) to reduce the size of the previous convolution layer
- In order to minimize overfitting in our model, we used multiple dropouts.
- We then flatten the image into a single dimension and used 3 dense layers to classify the image based on the output .
- The last dense layer was set to 4 nodes as it corresponds to the 4 possible categories the MRI image that it could fall under.
- We then optimized the model using adam and calculated our loss using categorical loss cross entropy.

Description of implementation

- To solve this problem, we first uploaded our dataset to Google Drive. From there we used the drive.mount function to connect the Drive to the file. From here we imported the data from each category into arrays, and recorded the associated labels
- We then normalized the data so it was converted from 0-255 to 0-1. This allows for the input data (pixels) to have a similar scale. Allowing for faster convergence while training the model. Also, we used hot encoding to deal with our labels.
- We then made use of our model to analyse the images. We then print out the accuracy and loss for both the training and test data. At the end we printed out a chart of the loss and accuracy over the epochs.
- We decided to use a split of 60% for training, 25% for testing, and 15% for validation. With this split we gave the model adequate data to formulate the best solution. While also leaving enough test data to allow the loss function to choose the most accurate model.
- We tested out the model by randomly selecting an image from the validation data and compared it to prediction value which was generated from the model.
- We also calculated the accuracy score over the entire validation set to ensure correctness of our model.



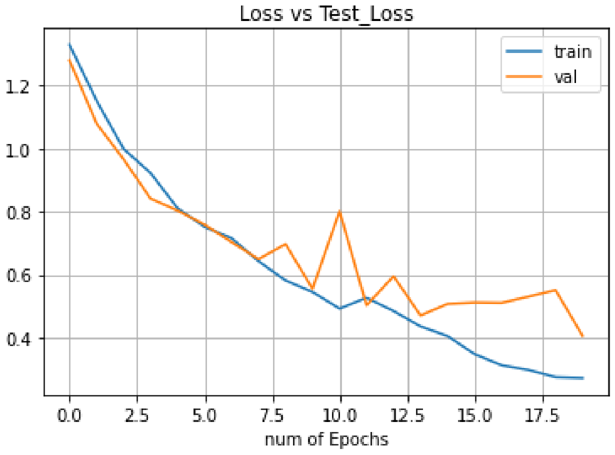
Summary of evaluation and Conclusion

- In order to ensure that the model runs with new test and training sets every time, we combined the test and training folders and divided them ourselves.
- To conclude, we used a dataset with over 3000 images of varying diagnoses of brain tumors. Using a convoluted neural network, we were able to design a model which accurately predicted whether the input MRI scan had a tumor and if so which kind of tumor it is.
- After training the model, we began validation testing. We used the X_val and Y_val datasets and compared them to the model’s prediction.
- During this testing, we yielded a result of 86.7% confidence. This was using data the model had never seen before. The model was successful in the task it was created for.

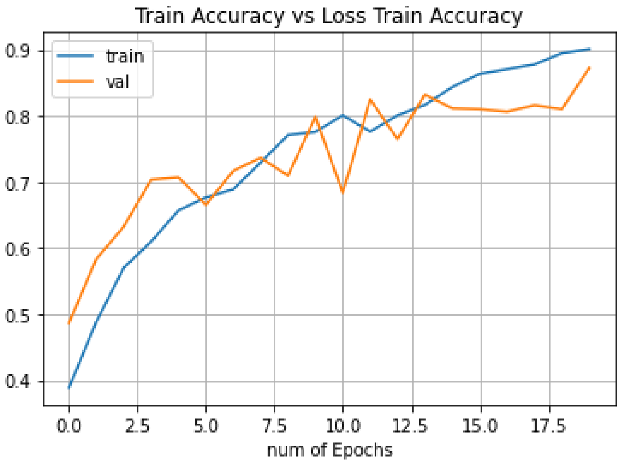
	Training	Testing	Validation
Loss	0.27	0.40	X
Accuracy	90.09%	87.25%	86.7%

Summary of evaluation and conclusion of project

This graph shows a steady decrease in the loss for both the training and test set.



This graph shows a steady increase in the accuracy for both the training and test set.



Based on these graphs, we depict that our model is neither overfitting or underfitting to our training model