Brain Tumor Classification in MRI Images Using Convolutional Neural Network

Final Report

Derya Kurt

June 2021

Table of Contents

1. INTRODUCTION

1.1 Problem Statement

Brain tumor is the collection is an abnormal mass of tissue in which cells grow and multiply uncontrollably, seemingly unchecked by the mechanisms that control normal cells. Brain tumors can be both malignant or benign. The most common types of brain tumors are Glioma, Meningioma and Pituitary tumor.

Meningioma is the most common benign tumor, comprising 10 to 15 percent of all brain neoplasms, although a very small percentage are malignant.

Pituitary is the most common tumor after gliomas, meningiomas and schwannomas. The large majority of pituitary tumors are benign and fairly slow-growing. Even malignant pituitary tumors rarely spread to other parts of the body.

Glioma is the most common primary malignant brain tumors in adults.

There are several medical imaging techniques used to acquire information about tumors (tumor type, shape, size, location, etc.), which are needed for their diagnosis. Magnetic Resonance Imaging (MRI) is one of the most used technique. These images are examined by the radiologist. A manual examination can be error-prone due to the level of complexities involved in brain tumors and their properties.

The goal of this project is to develop a Deep Learning approach by using Convolutional Neural Network and Transfer Learning (TL) to detect and classify Brain Tumor.

2. DATA AND PREPROCESSING

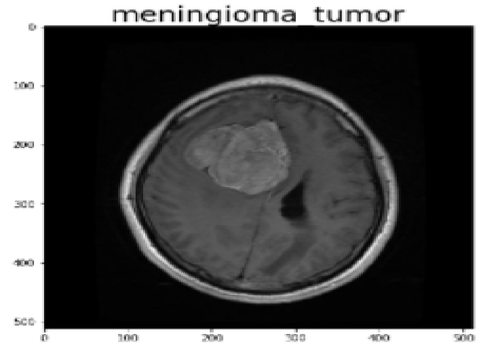
2.1. Data Overview

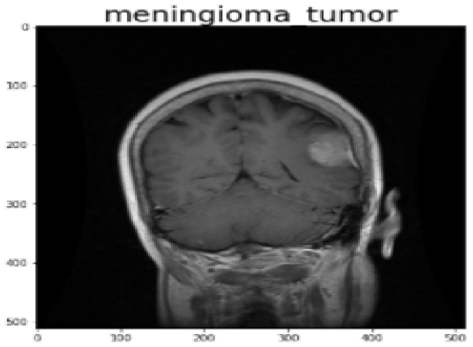
The data set provided by https://www.kaggle.com/sartajbhuvaji/brain-tumor-classification-mri is used in this study. The dataset contains 4 subfolders which are consisted of respective tumor classes, Glioma, Meningioma, Pituitary, and No Tumor datasets include 826, 822, 827 and 395 MRI images respectively.

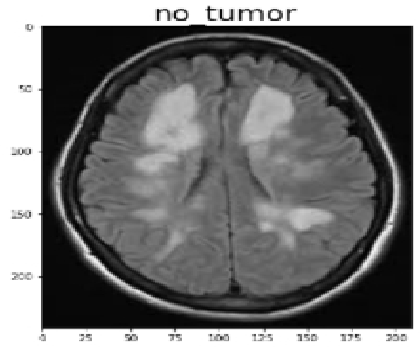
2.2 Data Preprocessing

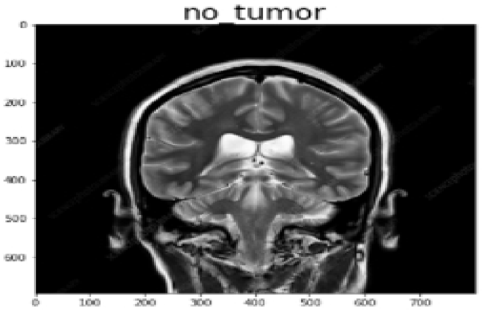
2.2.1 Image Processing:

Each tumor sub-folders are gathered into 1 data set. As shown below the images had different shapes with 3-channel.





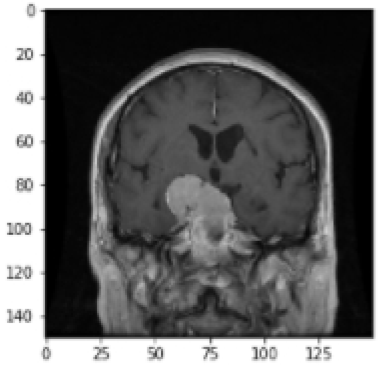




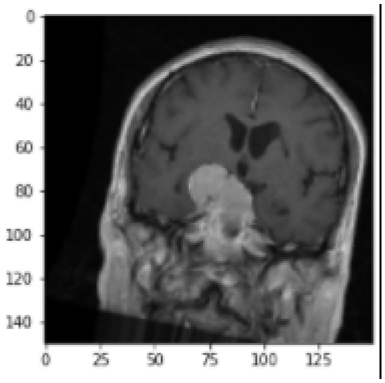
Therefore the images are resized to 150 x 150 and changed from RGB to GrayScale. The pixel values are normalized to [0,1] range by multiplying with 1/255. The class labels are turned into one-hot encoded binary matrix by encoding string to integers with LabelEncoder class and then converting the vector of integers to one hot encoding with to_categorical() function. Finally the data is splitted into training, validation, and testing having 2324, 259 and 287 respectively. The training data is for model learning, validation data is for model evaluation and model parameters tuning. Test data is for the final evaluation of our model.

2.2.2 Data Augmentation:

Since the data set was small in order to avoid overfitting in CNN Model 3 data augmentation is applied on the training data set and the original images are replaced with the augmented batch. During transformation modifications such as zooming, rotating, and shifting are made to the images. By this method we increased the diversity of the training set.



Original Image



Augmented Image

3. CLASSIFICATION METHODS AND MODELS

In this project I worked on 3 different approaches. Multi-Class Classifiers Method, Multiple Binary-Classifiers Method and Majority Voting Ensemble Method.

3.1 Multi-Class Classifiers Method:

3.1.1 Building the Model:

In Multi-class classification CNN models which have strong capability to capture spatial relationships of pixels across rows and columns and pre-trained models VGG-16 and RESNET50 are performed.

CNN Hyper Parameters:

Filter: helps to extract different features from an image like edges, vertical lines, horizontal lines, bends, etc. and generate feature maps.

Kernel_Size: width X height of the filter mask

Padding: adds zeros to the rows and columns to ensure both the output tensor and input tensor should have the same height and width

Activation: decides whether a neuron should be activated or not by enabling the back-propagation by updating the weights and biases and adds non-linearity to the output. **Relu**: performs a threshold operation on each input element where all values less than zero are set to zero and eliminates the vanishing gradient problem observed in sigmoid and tanh functions. **Softmax**: returns probabilities of the inputs as output and is used preferably for multivariate classification

Pooling: reduces the dimensionality of the feature map to prevent overfitting and improve the computation speed. **MaxPooling:** pooling operation that calculates the maximum, or largest, value in each patch of the feature map.

Stride: number of steps we take when sliding our window (filter) across an image.

Flatten: since fully connected layers accept only 1 Dimensional data. Flatten converts 2D data to 1D

Dense: adds the fully connected layer to the neural network by adding every input to every output by a weight

Dropout: random portion of the outputs are ignored during training to avoid overfitting (strong dependencies between portions of adjacent layers)

Optimizer: used to minimize the loss function by adjusting the weights and learning rates. **Adam Optimizer:** stochastic gradient descent method that computes individual adaptive learning rates for different parameters from estimates of first- and second-order moments of the gradients.

3.1.1. 1st Model: The Base CNN Model without Data Augmentation:

To see the effect of Data Augmentation on the prediction accuracy the CNN model is built without Data Augmentation preprocessing.

The Base CNN Model Architecture:

Model: Sequential layer which allow to build a model layer by layer is created.
Input Layer: the shape of data is defined as 150x150 with Grayscale

1st Convolutional Layer: applies 64 filters with kernel size 3x3 and ReLU activation function is used to add non-linearity.
1st MaxPooling Layer: takes window with the size of 2x2 array, returns the maximum number inside it and moves the window by 2 pixels
2nd , 3rd , 4th and 5th Convolutional Layers: applies 128 filters with kernel size 3x3 with ReLU activation functions
2nd , 3rd , 4th and 5th MaxPooling Layers: takes 2x2 filter and stride of 2

Fully Connected Layer:
Flatten:
Dense Layer: constructs dense layer with hidden layer and 512 nodes with ReLU activation function

Output Layer:
Dense Layer: number of classes which is 4 with softmax function.

Model Compiling:

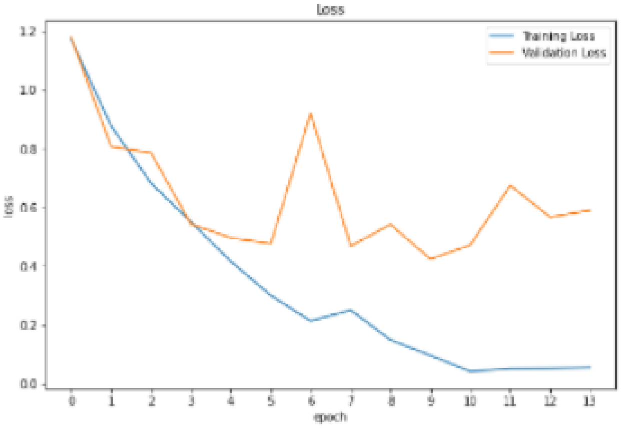
optimizer: Adam
learning rate: 0.001
loss: categorical_crossentropy (the output is probability distribution of the class over "n" classes and its sum is one. This means the probability for a class is not independent from the other class probabilities.)
metrics: accuracy (Calculates how often predictions equal labels.)

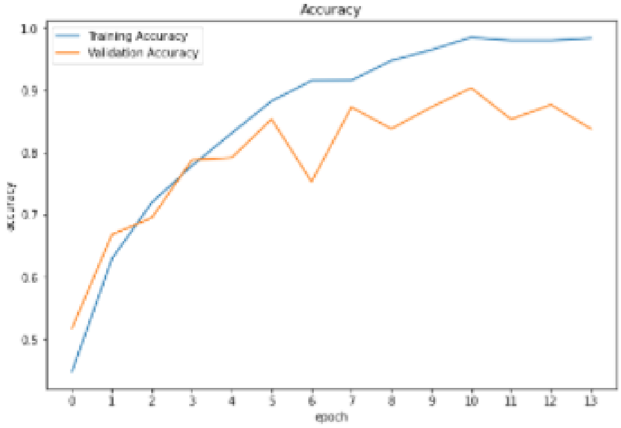
Model Training:

Callback: performs actions at various stages of training.
EarlyStopping: stops training when the monitored metric 'val_accuracy' (validation set accuracy) stops improving after the number of epochs (patience=3) with no improvement and for val_accuracy the mode should be 'max'
ModelCheckpoint: save the model weights at the maximum of the monitored quantity.
batch_size, epochs : 40, 20 There are 2324 images in the train data set. 2324 is divided into the batch size (40) which yields 59. And the algorithm takes the first 40 samples from training data and trains the networks. Next it takes the second 40 samples and trains. This repeats 59 times to complete for the whole train data set. At each epoch the entire data set is passed forward and backward through the network and weights are updated. With 20 epochs this will repeat for 20 times to find the optimal loss and accuracy.

Model Evaluation:

Although the model has 0.8378 Validation Accuracy and 0.9268 Test Accuracy, the below loss plot indicates the validation error is higher than training loss which means the model is not able to generalize to new data. That is due to overfitting.





3.1.2. 2nd Model: The Base CNN Model with Padding and Dropout without Data Augmentation:

The Model Architecture:

The below parameters are added to Base CNN Model.

DropOut: with rate of 0.2 to set 20% of out put to 0. Since training starts in the 1st layer Dropout is added in this layer to avoid the overfitting.

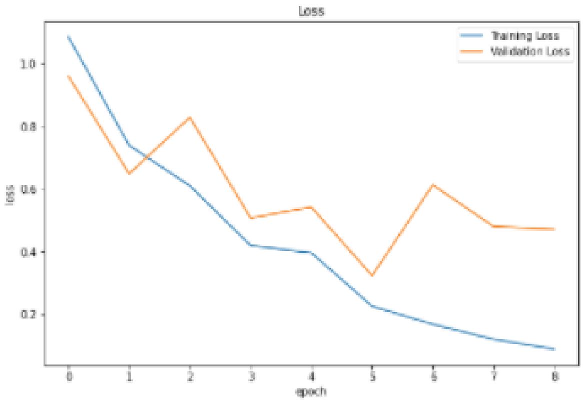
Padding: added in all Convolutional and MaxPooling Layers

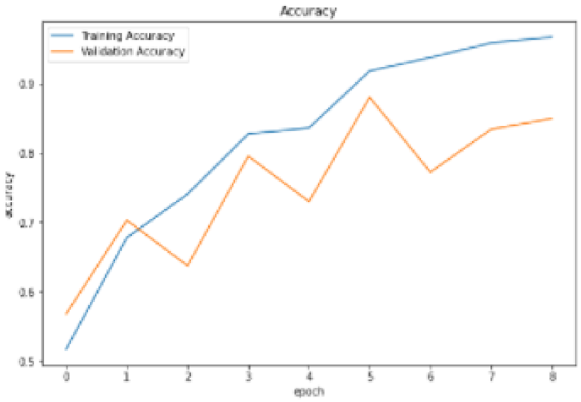
Model Compiling and Model Training:

The parameters in compiling and training remained same.

Model Evaluation:

The model has 0.8494 Validation Accuracy and 0.8885 Test Accuracy. Although the accuracy scores are high the below loss and accuracy plots show the 2nd Model is also overfitting.





3.1.3. 3rd Model: The Base CNN Model with Padding and Dropout and with Data Augmentation:

The Model Architecture:

ImageDataGenerator is constructed for data augmentation.

The images that are not in the center the pixels of the image are shifted. For a vertical shift of image **height_shift_range** and for a horizontal shift of image **width_shift_range** are set to 0.10

zoom_range is set to 0.2 which means to zoom in and zoom out 20%. So the range for the zoom will be [0.8,1.2] Zoom in makes the object larger, whereas zoom out makes smaller.

horizontal_flip is set as True and **shear_range** is set to 0.1 which stretchs the image 10%

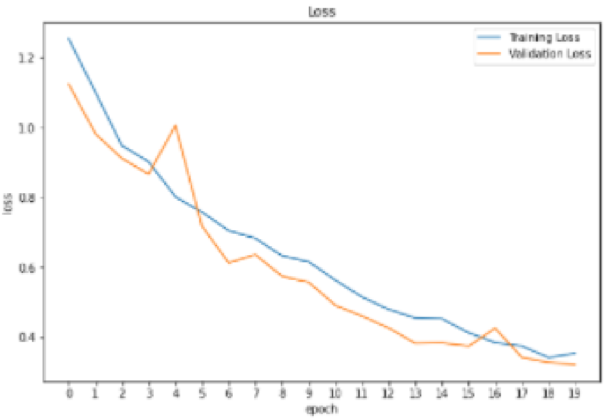
With the **fill_mode**='nearest' parameter enables the empty area in the image to be replaced with the nearest pixel values.

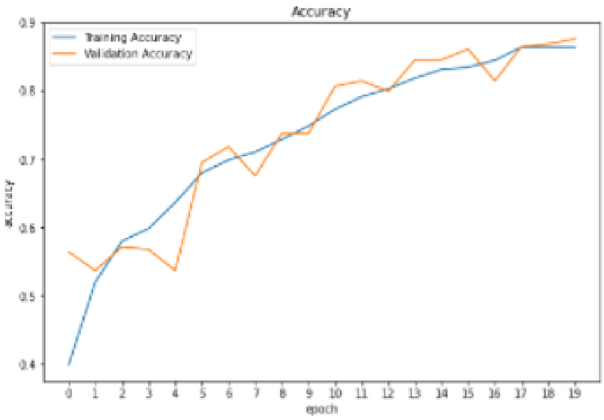
Model Compiling and Model Training:

The parameters in model architecture, compiling and training remained same.

Model Evaluation:

The model has 0.8764 Validation Accuracy and 0.9059 Test Accuracy. By applying data augmentation the model becomes more robust and avoids the overfitting. However the loss of accuracy is higher than loss of validation. This may be due to regularization method dropout. So in the next model I worked with the same model without Dropout.





3.1.4. BEST MODEL: Model_Multiclass - The Base CNN Model with Padding and Data Augmentation:

The Model Architecture:

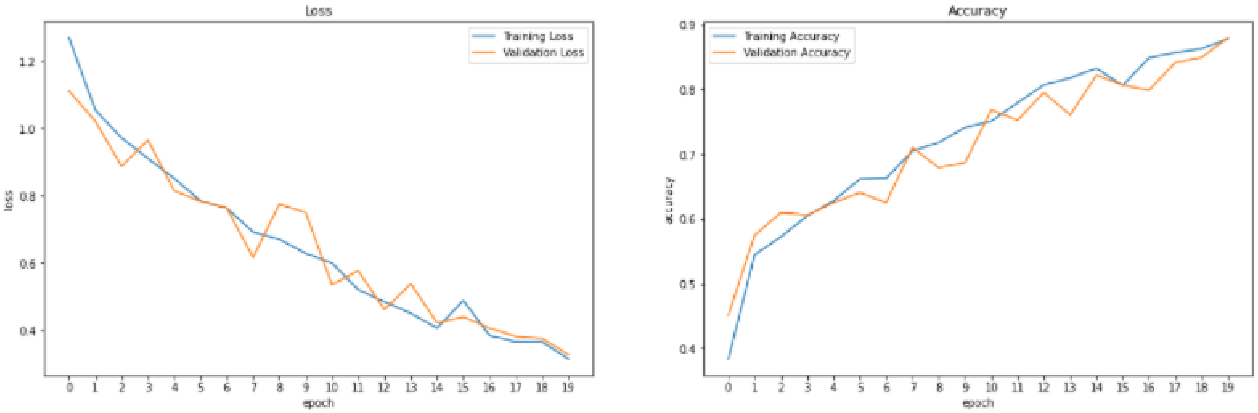
This model has the same structure with the previous one (Model 3) but without Dropout.

Model Compiling and Model Training:

The parameters in model architecture, compiling and training remained same.

Model Evaluation:

The model has 0.8803 Validation Accuracy and 0.8815 Test Accuracy. This is the best model among all the models.



TRANSFER LEARNING MODELS:

3.1.5. 5th Model: VGG16

Model Architecture:

VGG16 is a pre-trained model which takes RGB images with default input size (224,224). However I changed the input size as (150,150,3) and applied data augmentation.

In order to use pre-trained weights the **weights** are set as **"imagenet".** To exclude the default classifier with 1000 classes in FCL (Fully Connected Layer) **include_top** is set as **False**.

In the next step the layers in the VGG16 Model are set as not trainable in order to not be updated during backpropagation process in training.

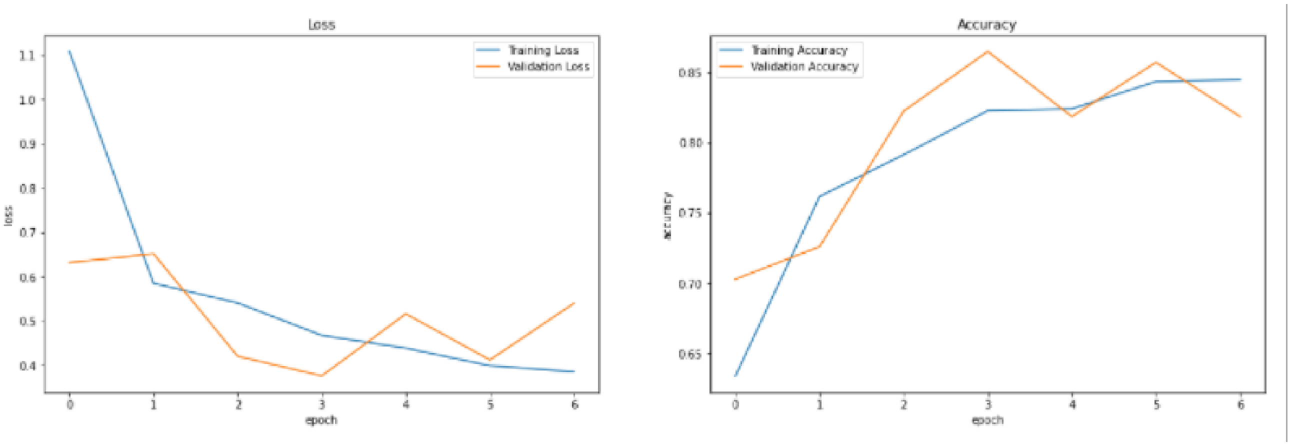
And finally I built and attached the last fully-connected layer by flatting the last layer and adding 4 neurons and softmax activation function.

Model Compiling and Model Training:

The parameters in model architecture, compiling and training remained same.

Model Evaluation:

The model has 0.8185 Validation Accuracy and 0.8711 Test Accuracy. The model is not stable. Since the accuracy didn't improve after the 6th epoch the training stopped.



3.1.6. 6th Model: RESNET50

Model Architecture:

More layers slow the training and accuracy can get worse instead of improving since having deep CNN causes vanishing gradient problem. The reason is during backpropagation to the initial layers derivative becomes almost insignificant value specially for small weights.

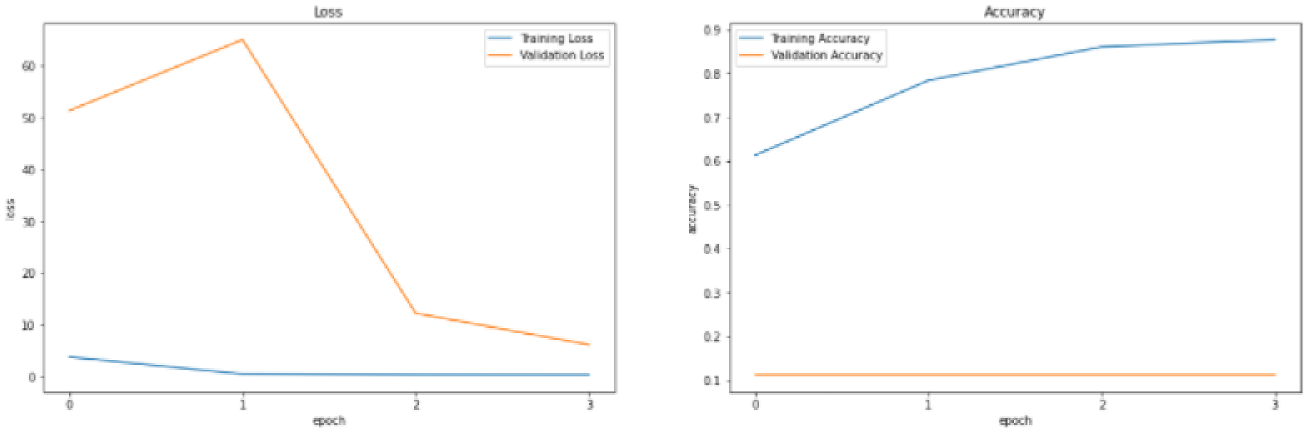
In Residual Network (ResNet) the Skip Connections between layers add the outputs from previous layers to the outputs of stacked layers.

Model Compiling and Model Training:

The parameters in model architecture, compiling and training remained same.

Model Evaluation:

The model with 0.115 Validation Accuracy and 0.115 Test Accuracy it has very poor performance.



3.2 Multiple Binary-Classifiers Method:

In this method, the training data set is divided into 3 binary classes. In each binary classes, the data set is splitted into training, validation, and testing data sets.

Besides the number of epochs all binary classifiers has the same parameters in compiling and training with multi-class models. Number of epochs is changed as 50.

3.2.1. MENINGIOMA Binary Classifier:

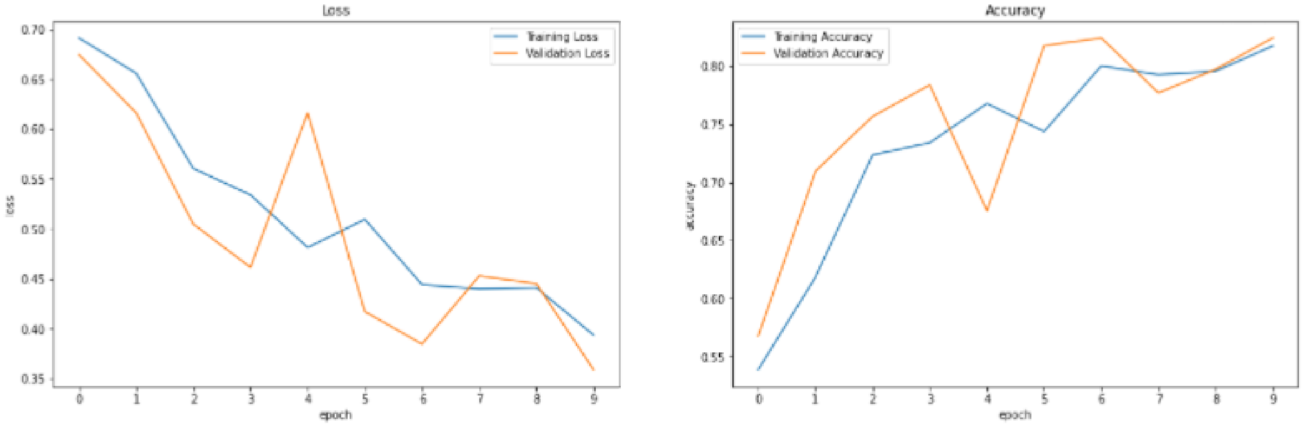
This binary classification has 2 classes: **Meningioma & Non-Meningioma.** The data set consists of 822 images of Meningioma, and 822 Non-Meningioma images including 411 images from each Pituitary and Glioma data sets.

Model_Meningioma Architecture:

Data augmentation is applied and the base model with padding and Dropout (0.1) has the best result.

Model Evaluation:

The model has 0.8243 Validation Accuracy and 0.8061 Test Accuracy.



3.2.2. GLIOMA Binary Classifier:

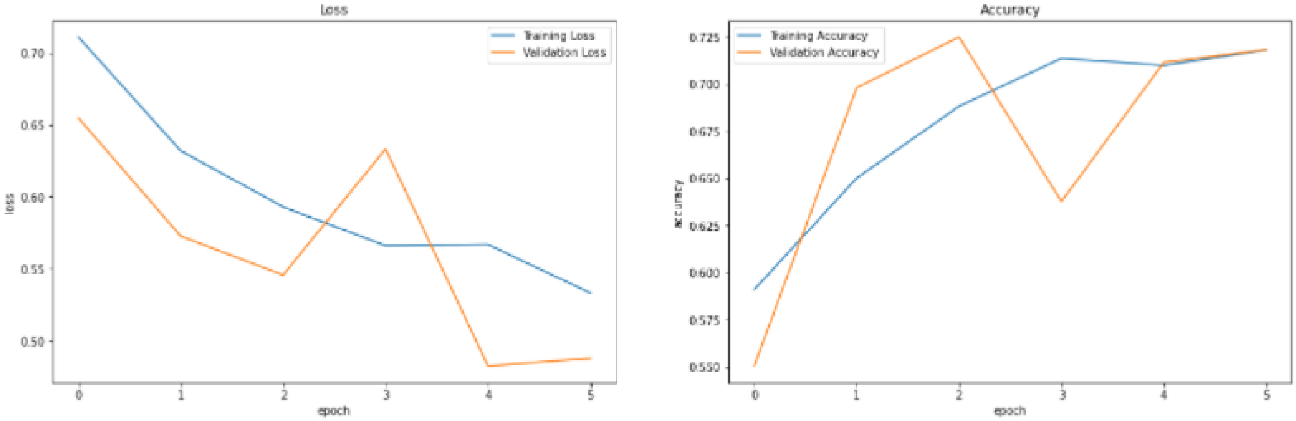
This binary classification has 2 classes: **Glioma & Non- Glioma** The data set consists of 826 images of Glioma, and 822 Non- Glioma images including 413 images from each Pituitary and Meningioma data sets.

Model_ Glioma Architecture:

Data augmentation is applied and the model with (1 input, 3 convolutions, 3 max pooling, 1 fully connected, 1 dropout(0.1), and 1 classification) layers has the best result.

Model Evaluation:

The model has 0.7181 Validation Accuracy and 0.7289 Test Accuracy. Since the accuracy didn’t improve after the 5th epoch the training stopped.



3.2.3. PITUITARY Binary Classifier:

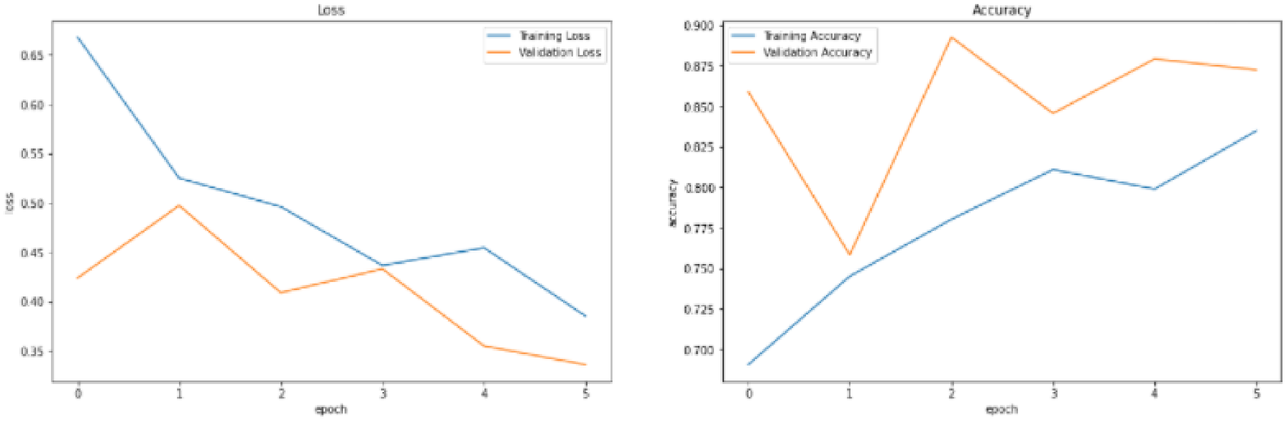
This binary classification has 2 classes: **Pituitary & Non- Pituitary** The data set consists of 827 images of Pituitary, and 826 Non- Pituitary images including 413 images from each Glioma and Meningioma data sets.

Model_Pituitary Architecture:

The structure of the model is same with the Glioma Model.

Model Evaluation:

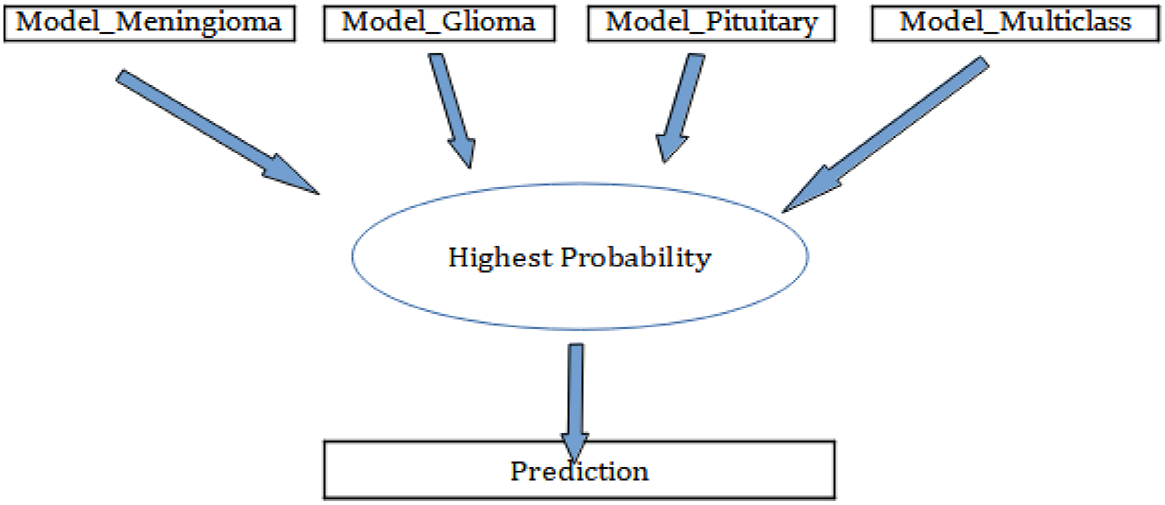
The model has 0.8725 Validation Accuracy and 0.8675 Test Accuracy.



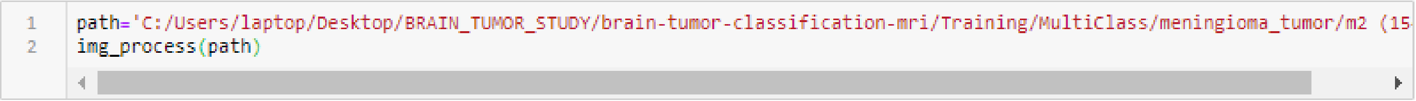
3.3 Ensemble Method:

In this section the ensemble method which uses multiple learning algorithms to obtain better predictive performance than could be obtained from any learning algorithms alone, is applied by concatenating the best binary classifications models and Model_Multiclass based on the highest probability method.

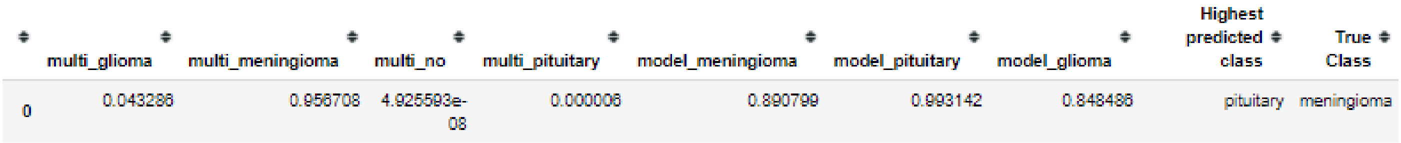
For each image the probability of belonging to a class is calculated and the model with the highest prediction probability is chosen to be where the image belongs.



The below table indicates the prediction probabilities of Meningioma tumor image for each class.



```
path='C:/Users/laptop/Desktop/BRAIN_TUMOR_STUDY/brain-tumor-classification-mri/Training/MultiClass/meningioma_tumor/m2 (15
img_process(path)
```



	multi_glioma	multi_meningioma	multi_no	multi_pituitary	model_meningioma	model_pituitary	model_glioma	Highest predicted class	True Class
0	0.043286	0.956708	4.925593e-08	0.000006	0.890799	0.993142	0.848486	pituitary	meningioma

Multiclass model predicts the probability of image to be glioma 0.04, meningioma 0.96, whereas pituitary and no tumor very small. **Model_meningioma** predicts the probability of image to be meningioma as 0.89. According to **model_pituitary** the probability of being pituitary is 0.99, whereas **model_glioma** predicts its probability to be glioma as 0.85. Among these classification The highest prediction probability (0.99) belongs to model_pituitary which classified the image as pituitary. Therefore **Ensemble Model** based on the highest probability classifies our image as "pituitary".

4. Metrics and Evaluation:

The goal of this project is to detect and classify the tumor correctly. Capturing the tumor is highly important for treatment. Therefore recall score which measures the model in identifying the True Positives correctly is an important decision criteria in model evaluation.

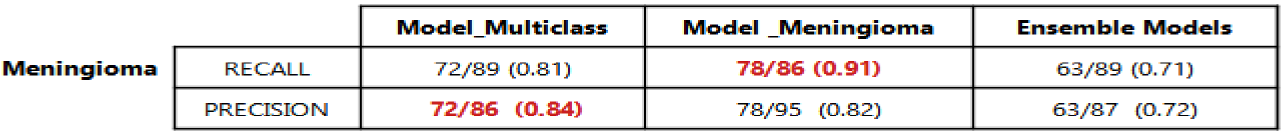
Since the ratios of tumor in multi-class and binary classes data sets are different, accuracy metric is not a good option for model evaluation.

Although recall is crucial in tumor diagnosis and treatment, precision metric which is important to avoid improper treatment for a patient with no disease or different tumor disease, plays an important role in stakeholder’s decision. Therefore both recall and precision are calculated.

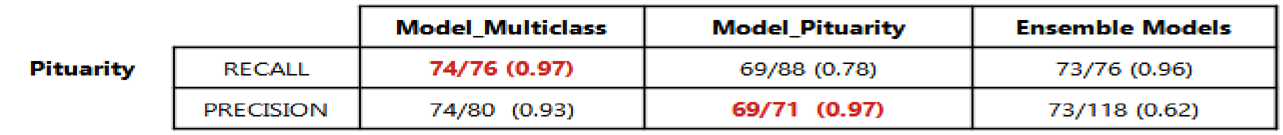
The confusion matrix provided further insight to see the actual and predicted positive and negative classes. By using confusion matrix recall and precision are calculated per each tumor type with the above best models.

Recall and Precision Results in Tumor Prediction:

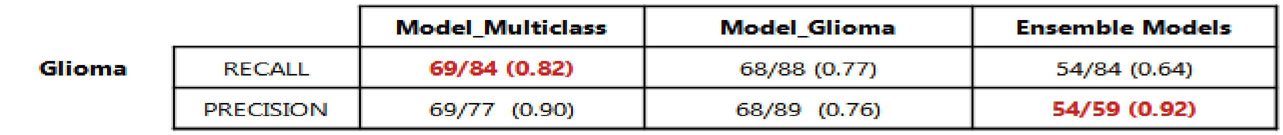
Recall = True Positives / (True Positives + False Negatives)
Precision = True Positives / (True Positives + False Positives)
The below chart indicates the recall and precision of each tumor per models.



		Model_Multiclass	Model _Meningioma	Ensemble Models
Meningioma	RECALL	72/89 (0.81)	**78/86 (0.91)**	63/89 (0.71)
	PRECISION	**72/86 (0.84)**	78/95 (0.82)	63/87 (0.72)



		Model_Multiclass	Model_Pituarity	Ensemble Models
Pituarity	RECALL	**74/76 (0.97)**	69/88 (0.78)	73/76 (0.96)
	PRECISION	74/80 (0.93)	**69/71 (0.97)**	73/118 (0.62)



		Model_Multiclass	Model_Glioma	Ensemble Models
Glioma	RECALL	**69/84 (0.82)**	68/88 (0.77)	54/84 (0.64)
	PRECISION	69/77 (0.90)	68/89 (0.76)	**54/59 (0.92)**

4.1 Meningioma Tumor Prediction:

Recall: Model_Meningioma is the best model in predicting the Meningioma tumor. It classified 78 out of 86 Meningioma tumor as Meningioma, while missed 8 Meningioma tumor images.
Precision: Model_Multiclass classified 72 out of 86 Meningioma tumor as Meningioma correctly, while 14 Non-Meningioma images as Meningioma.

4.2 Glioma Tumor Prediction:

Recall: Model_Multiclass predicted 69 Glioma correctly out of 84., while missed 15 Glioma tumor.
Precision: Ensemble Model classified 59 images as Glioma, where 54 were predicted correctly, remaining 5 images falsely.

4.3. Pituitary Tumor Prediction:

Recall: Model_Multiclass which classified 74 Pituitary tumor out of 76 is the winner model.
Precision: Model_Pituitary predicted 71 images as Pituitary. Although 69 of them were predicted correctly, the remaining 2 images were falsely predicted positive.

For each type of tumor we have high precision and recall. However Model_Glioma and Ensemble Model had poor performance in overall.

5. CONCLUSION

In this project 3 different approaches were presented to classify the brain tumors.

First, CNN models and Transfer Learning Models VGG-16, ResNet-50 are developed using different data preprocessing techniques on multi-class data set. Data Augmentation played an important role in avoiding overfitting. Validation-Test Accuracies and Training and Validation Loss-Accuracy plots are used to determine the best model. Our CNN model had higher performance than trained models.

Second, multi-class data is divided into 3 binary classification data sets with equal class distributions. For each binary data set the best CNN models are evaluated.

And finally Ensemble Model is performed by concatenating the best binary classification models and multi-class model. The classification method in Ensemble Model is based on the highest prediction probability. The tumor is likely to belong to class with the highest prediction probability.

Since minimizing the false negatives and false positives in tumor treatment is highly important precision and recall metrics are used to evaluate the models. In Meningioma Tumor classification Model_Meningioma has the highest recall 0.91, while Model_Multiclass has 0.84 precision. Pituitary Tumor classification has recall and precision scores 0.97 with Model_Multiclass and Model_Pituitary respectively. In Glioma tumor classification Model_Multiclass has recall 0.82 and Ensemble Model has

0.92 precision score. Each tumor class have the highest recall and precision results with different CNN models. But overall the results for each tumor classes are almost over 0.90.

For each tumor classification Multi-class CNN model has the highest score in either recall or precision. Additionally it has 0.8815 Test Accuracy. Therefore it is the best final model to work with.

6. RECOMMENDATION AND FUTURE WORK

Unfortunately the performance of Ensemble Method was below my expectation. Majority Voting Ensemble Method would be worked on to get better performance.

One of the challenges of the project was the lack of large data set. To overcome this problem data augmentation is applied. However, massive amount of data for training will enable to build robust models.

Due to limited memory and GPU constraints GridSearch method to improve the algorithms couldn’t apply for hyperparameter optimization. To overcome this issue pre-trained CNNs are deployed.

As a future work building more robust deep learning model by using more images and applying GridSearch by using GPUs would be helpful to improve the performance.

In conclusion despite the constraints outlined above, the deep learning models had high performance in brain tumor segmentation.