Tumor Detection Using Deep Learning

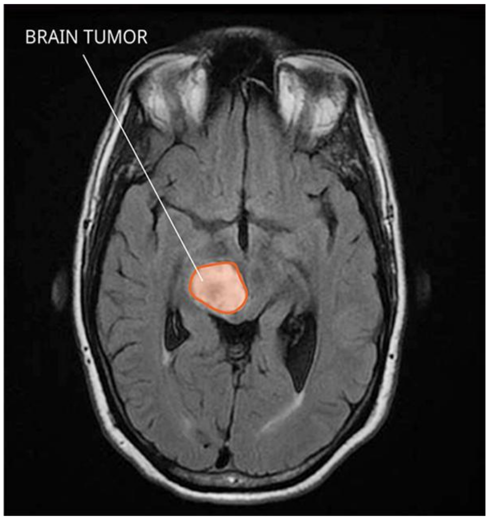
PROJECT REPORT

By

Harshal Gadhe

January 28, 2021

1. Introduction



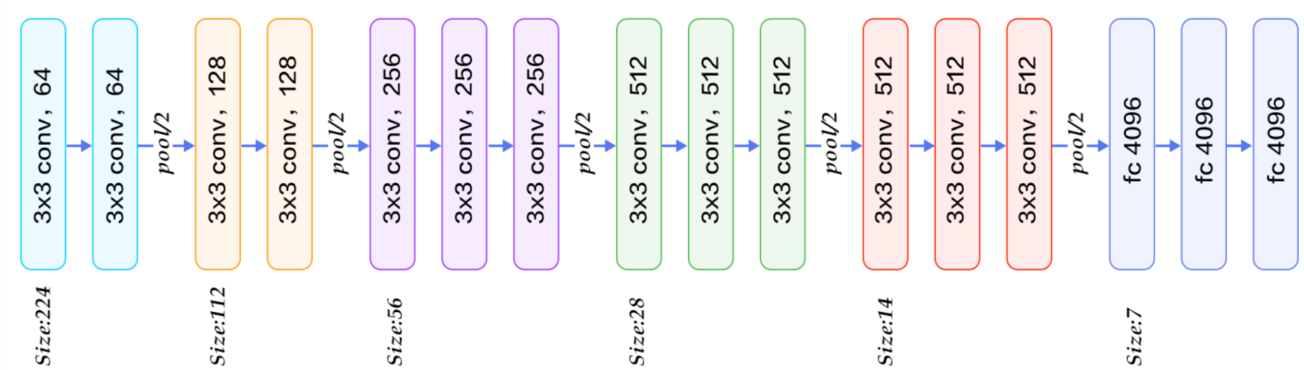
Due to excessive cell division tumor is formed which when not treated early can cause death of the patient. To solve this problem, we will be using Machine Learning so as to detect tumors from the MRI scan of the patient's brain. We would find the model having highest accuracy in detecting the tumor in the brain.

2. Approaches & Results

To train the Model on the MRI images which consists of 2 types of classes namely 'yes' & 'no' for the images containing Tumor and other with No Tumor respectively. We will build the model using TensorFlow and other deep learning libraries like keras, etc. To extract the features from the images CNN architecture along with other layers is used in the model. To increase model accuracy model, model was trained on images by Image Augmentation techniques on original images like rotation, brightness, flipping with the help of Image Data Generator.

Initially the model is trained on custom made layers but the result was inconsistency in accuracy and loss which made it harder for evaluating the model's performance. To avoid the above problem and increase the performance of model Transfer Learning was used to train the model. VGG16 architecture was used to train the model whose architecture is show in the figure below.

VGG16 Model Architecture

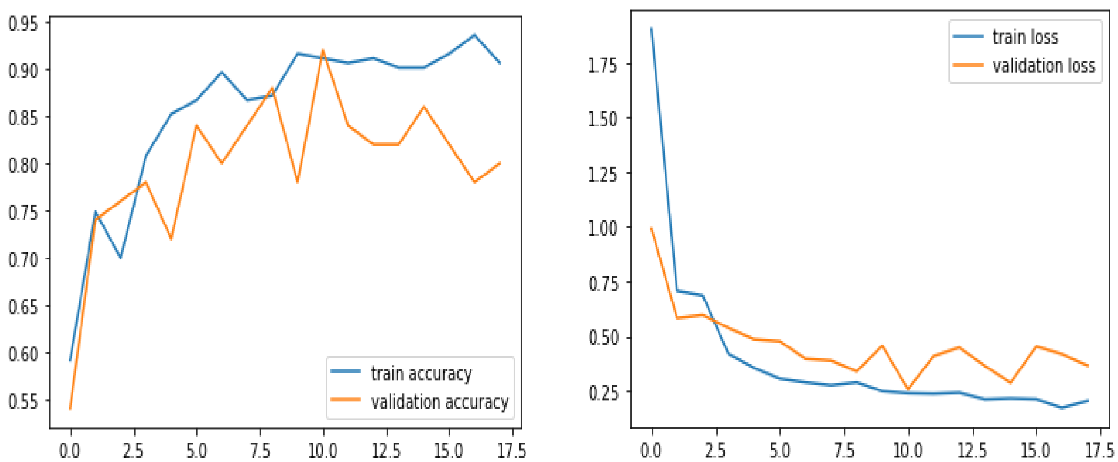


As we need to classify the images into 'Tumors' and 'Normal' we will remove the last three layers from the architecture and freeze rest of the layers to conserve the weight before we add layers like Flatten, Dropout and Dense layers in the model. During training the model, it was found that a Dense layer consisting of 256 gave the best possible outcome while training.

Callbacks such as 'Early Stopping' and 'Model Check Points' are used so as to save the best weights and stop the model earlies once we get the desired result. These callbacks monitor the validation accuracy while training the model and stop the training when was no increase in the validation accuracy for consistently 6 epochs (as patience was set to 6).

The table below shows the performance of the model after it is trained. We used 30 epochs for training the model but the model was train on 18 epochs due to early stopping.

Accuracy and Loss of Model vs Epochs



Model Performance of model after training on VGG16 architecture

Training		Validation	
Loss	Accuracy	Loss	Accuracy
0.2044	0.90~0.94	0.3659	0.82~0.86

3. Conclusion

Transfer Learning is the best option found while training the model than custom made model. Use of Dropout in the model resulted in solving overfitting to some extent by decreasing the variance. It was found that accuracy can be further improved by increasing the dataset size.