15CSE380 – NEURAL NETWORKS AND DEEP LEARNING

FINAL PROJECT – BRAIN TUMOR CLASSIFICATION

Executive Summary:

This project is done as a solution to the dataset which is provided by Kaggle. This project is an image classification problem using deep Convolutional Neural Networks to classify whether the image of a brain MRI has a tumor or not. Better performance of this model would help the medical professionals to detect tumors in MRI scans where the number of images is very high and manual processing is time taking.

Problem Description :

This problem is to classify an MRI scan image of a brain, based upon whether it has a tumor or not. There are two classes in which the images have been classified i.e, ‘yes’ and ‘no’, we need to build a deep learning model such that, given a new image of brain MRI, the model should be able to classify it to a class ‘yes’ or ‘no’ with a reasonably high accuracy measures.

The existing way to address this problem is to use a VGG-16 model which gave accurate results with the ‘Imagenet’ dataset. Some other models which were used are Vgg19 etc..

Description About Dataset :

The dataset was available in Kaggle in the datasets section. Here is a link to it <https://www.kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection>

There are 253 images in total, out of which 168 images were in the ‘yes’ class, and the rest were classified as ‘no’. The dataset is 15 MB in size and the images were of jpg and jpeg formats

‘Yes’ - encoded as 1, contains the tumor.

‘No’ - encoded as 0, contains no tumor.

Approach:

There are many ways to solve this problem and the traditional way is to use VGG-19 models.

A VGG model is one of the most powerful and complex Convolutional Neural Network(CNN) which can give highly accurate results in image classification. So we used a concept called transfer learning, through which we can use a pre-trained model(which was previously trained on another dataset) and can add more layers to it, and then fit the model training data. But before we can do this we need a properly preprocessed dataset to work with.

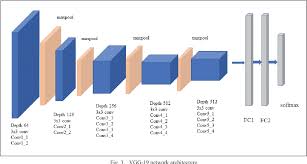
As we should also consider alternatives for the existing approach, we also used CNN’s like Resnet 50, VGG-16 models and chose the model with the best test accuracy.

Steps taken in the preprocessing step are :

1. Create three directories named ‘TRAIN’, ‘VAL’, ‘TEST’ in order to partition the dataset.
2. Split the dataset 80% of the data for training(‘TRAIN’), 5 data points for testing(‘TEST’), and the remaining for validation (‘VAL’).

Developing a VGG-19 model :

Pretrained model is available in Keras framework. We import that base model and input images that are resized to (227,227), but before inputting, we added fully connected layers and the output layer is of size two for classification. An important thing to notice here is while we are training the data, we do not have to train the VGG-19 model layers again. So we disabled training for those layers i.e, only the fully connected layers will be trained.



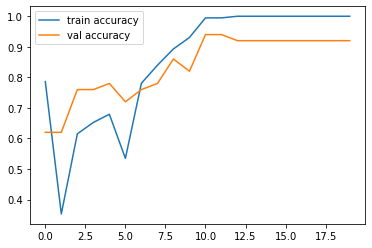
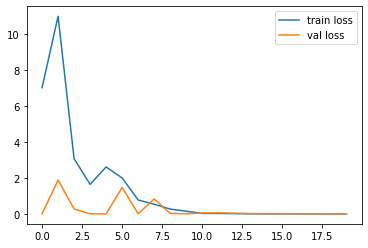
Data Augmentation :

The need for data augmentation here is to artificially increase the training data set size. Thankfully ImageDataGenerator from Keras does this for us. We need to pass in the vgg\_19\_preprocess input as a preprocessing function so that the augmented data images will be ready to be inputted to the VGG-19 model. And now this instance of ImageDataGenerator will take input from ‘TRAIN’ directory through flow\_from\_directory function and the training data is ready to be trained by the model.

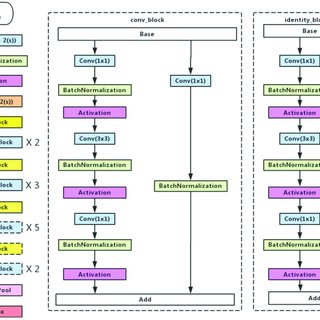
Training and Validation Phase:

**Here the loss value plotted is the sum of all errors(predicted - target) per epoch.**

Given are the plots for loss and accuracy for the two partitions.



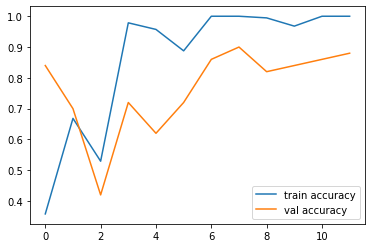
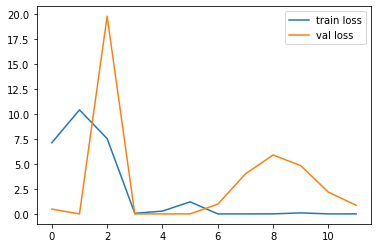
RESNET50 :  
 Resnet50 are one another such CNN which gives good results with image classification. The preprocessing and data augmentation were done using Resnet50 preprocess function and the model was fit to the training data



Training and Validation Phase:

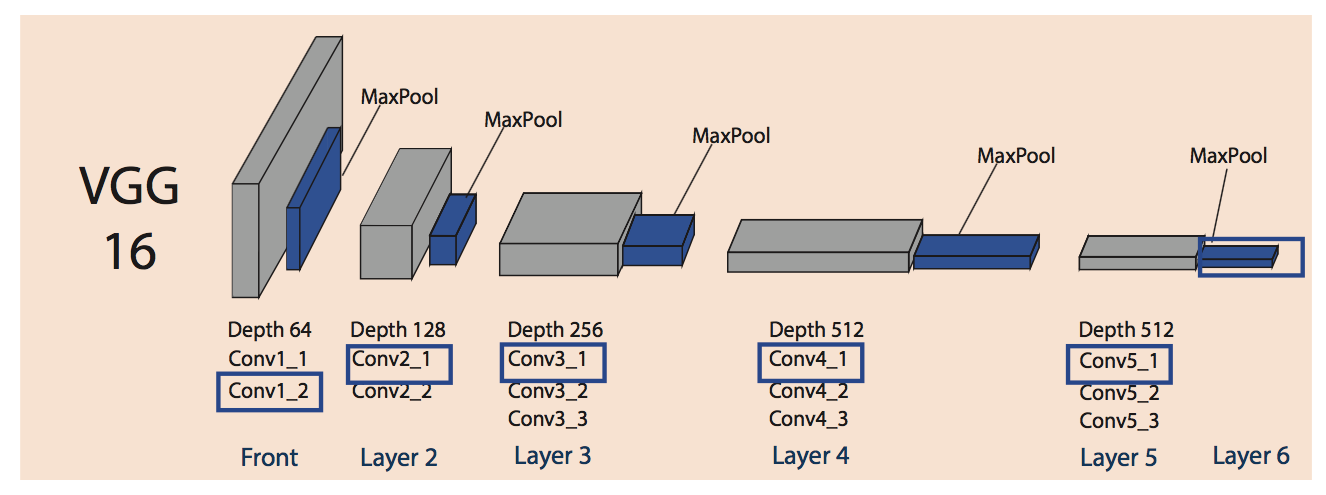
**Here the loss value plotted is the sum of all errors(predicted - target) per epoch.**

The following are the plots of the Loss and Accuracy values for the two partitions.



VGG-16:

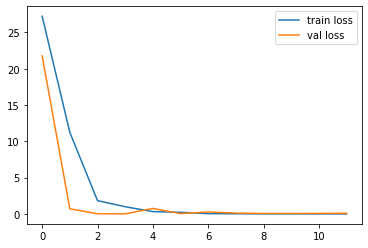
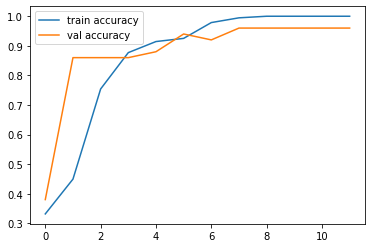
VGG-16 is the previous version of the VGG-19 model. It is also as powerful and accurate as VGG-19. The preprocessing and data augmentation were done using vgg16 preprocess function and the model was fit to the training data



Training and Validation Phase:

**Here the loss value plotted is the sum of all errors(predicted - target) per epoch.**

Given are the plots for Loss and Accuracy values for the train and validation partition.

Key findings and results of the model on Test data:

The VGG-16 model has the best accuracy value for the test dataset which around 93% (i.e it is able to classify 93 samples out 100)

Followed by VGG-19 which has the test\_accuracy value of 81%

Followed by RESNET 50 which has the accuracy value of 75%

I.e, given a new image the VGG-19, VGG-16 models are more likely to classify it correctly.

Conclusion:

This project was a combination of the CNN model classification problem (to predict whether the image has a brain tumor or not) & Computer Vision problem (image processing and image augmentation). The final accuracy is much higher than the 50% baseline. However, it could be increased by a larger number of train images or through model hyperparameters tuning.

GROUP - 5

P.V.S.S ALAVANDAR (AM.EN.U4CSE17150)

P.VAMSI KRISHNA (AM.EN.U4CSE17152)

NISHANT SAI (AM.EN.U4CSE17149)