**Project Report**

**Group:** Saahil Anande, Do quan, Mukka himaneesh

**Title:** Image Segmentation on Brain MRI Images Using U-Net

**Introduction:**

* We are attempting use U-Net on Brain MRI images to apply image segmentation to isolate LGG tumor cells in the brain.
* This method will aid in Brain Cancer diagnosis and treatment planning.

**Data Set:**

* LGG segmentation dataset from Kaggle
* The dataset consists of Brain MR images together with manual FLAIR abnormality segmentations masks
* They correspond to 110 patients included in the cancer genome atlas (TCGA) - Lower Grade Glioma collection with Fluid-Attenuated Inversion Recovery (FLAIR) sequence
* The tumor clusters and patient data is provided in a CSV file

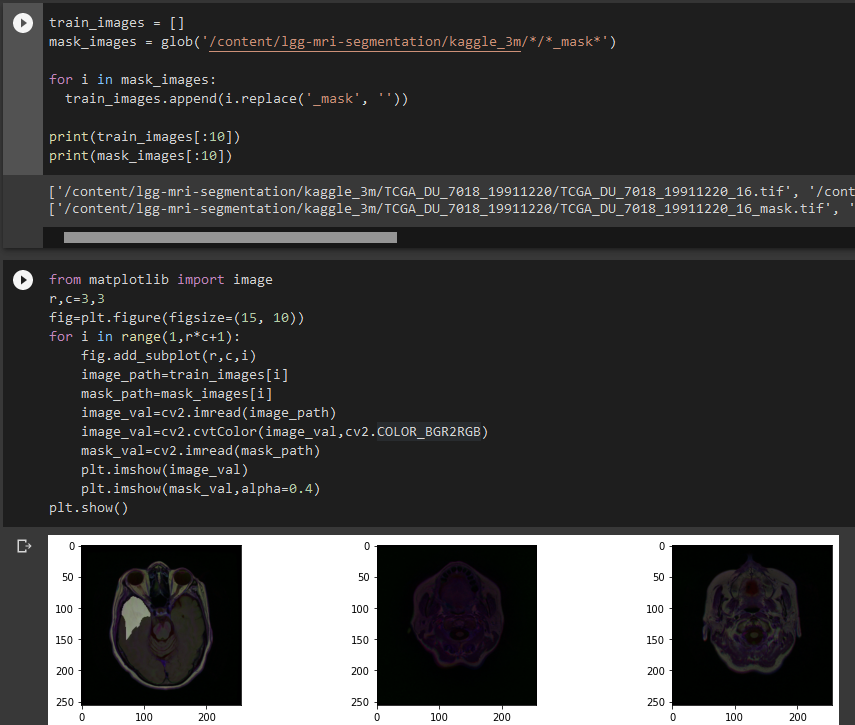
**Sample Images in the data set:**

Chart

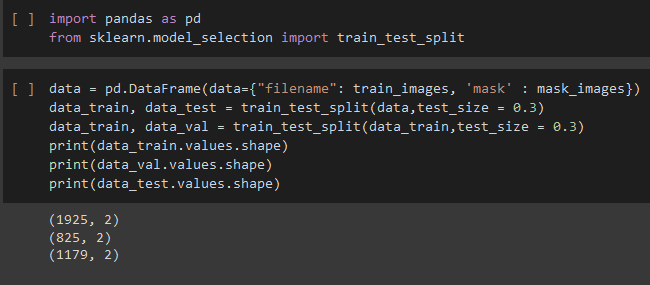
Description automatically generated

**Methodology:**

* **Loading the data set from Kaggle using the Kaggle API**



* **We then split the data set into Train, Validation and Testing**



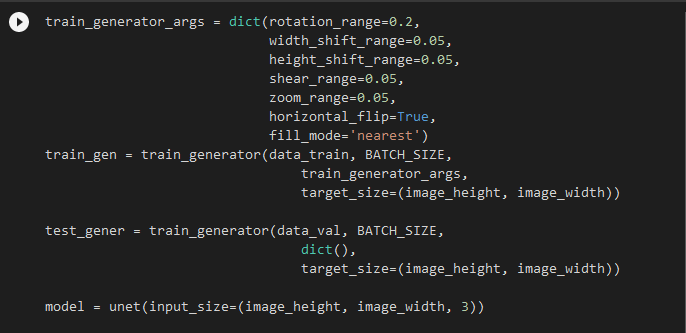
* Split ratio is 0.2, so 20% of the data will be Testing and rest will be Training and Validation

**Pre-Processing of the images:**

* Resized the images to 128x128 resolution, to reduce the load on our systems when training the model
* Applying the following augmentation methods:
* Rotation : Rotates the image by 20%
* Width Shift: Increase the width of image by 5%
* Height Shift: Increase the height of image by 5%
* Shear: Cutting and warping the image on the edges by 5%
* Zoom: Zooming on the image by 5%
* Flipping: Inverting the image horizontally

These augmentations were done to train our model better. We wanted our model to be able to track tumors on a range of warped images rather than just the vanilla data set.

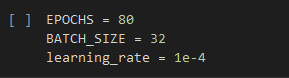
We also applied Thresholding of the images. The RGB pixel values were divided by 255 to give us a range of pixels from 0 to 1. Then taking a threshold value of 0.5, Values <0.5 were assigned 0 and Values > 0.5 were assigned 1. This resulted in the masked images being more prominent. This is because the focus will be on pixels having value of 1 hence all “White” areas will get the focus.



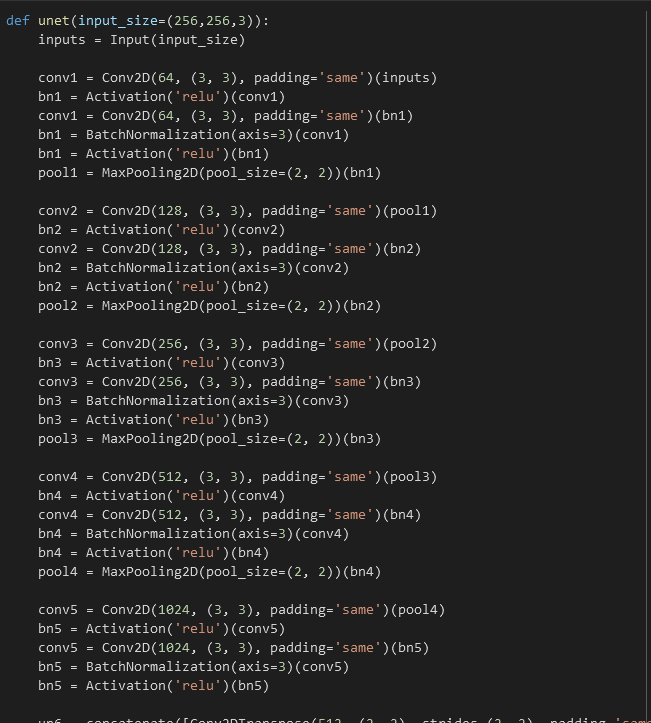
Here the image height and image width is 128 , 128

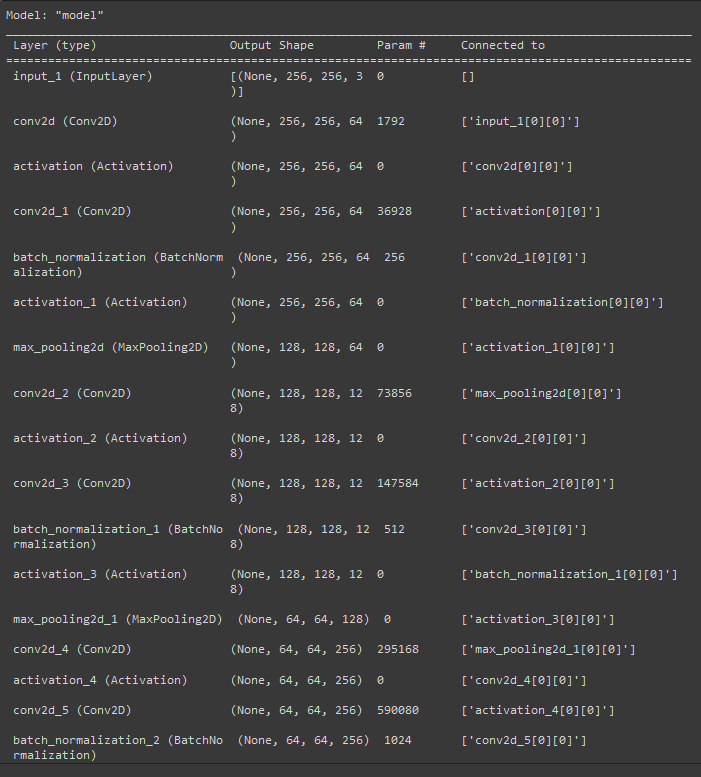
**Model Parameters used for training:**

* Gradient Optimizer: ADAM Optimizer
* Learning Rate: 0.0001
* Decay Rate(Adaptive Learning Rate for ADAM): 0.0001/80
* Batch Sizes for training: 32
* Iterations: 80



**Model Structure**: U-Net Layers





**Evaluation Parameters:**

**1) IOU (Intersection Over Union) :**

IOU(Intersection over Union) is a term used to describe the extent of overlap of two boxes. The greater the region of overlap, the greater the IOU.

IOU is mainly used in applications related to object detection, where we train a model to output a box that fits perfectly around an object. For example in the image below, we have a green box, and a blue box. The green box represents the correct box, and the blue box represents the prediction from our model. The aim of this model would be to keep improving its prediction, until the blue box and the green box perfectly overlap, i.e the IOU between the two boxes becomes equal to 1.

**A picture containing cat, mammal, domestic cat, gray

Description automatically generated**

**2) Jaccard Coefficient: T**he Jaccard similarity index (sometimes called the Jaccard similarity coefficient) compares members for two sets to see which members are shared and which are distinct. It's a measure of similarity for the two sets of data, with a range from 0% to 100%. The higher the percentage, the more similar the two populations.

**3) Dice Coefficient: It is 2 \* the Area of Overlap divided by the total number of pixels in both images.** The Dice coefficient is very similar to the IoU. They are positively correlated, meaning if one says model A is better than model B at segmenting an image, then the other will say the same. Like the IoU, they both range from 0 to 1, with 1 signifying the greatest similarity between predicted and truth.

4) **Loss Value**: Loss is a value that represents the summation of errors in our model. It measures how well (or bad) our model is doing. If the errors are high, the loss will be high, which means that the model does not do a good job. Otherwise, the lower it is, the better our model works.

Loss Graph:

**Chart, histogram

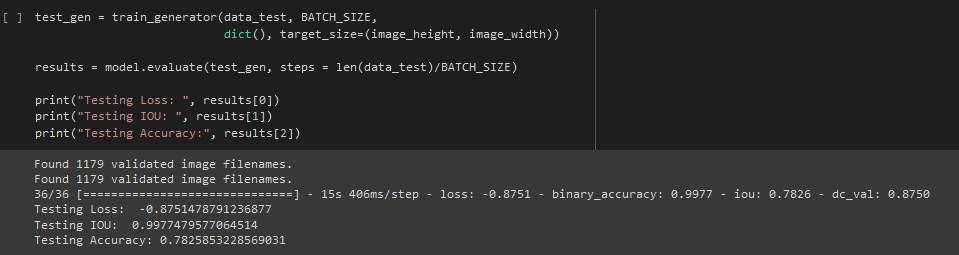
Description automatically generated**

**Accuracy Graphs:**

**Chart, line chart, histogram

Description automatically generated**

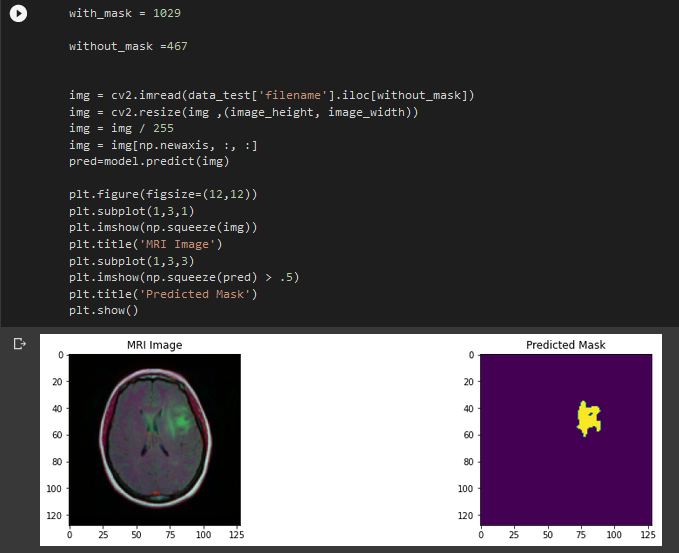
**Testing Results:**

****

**Graphical user interface, application

Description automatically generated**

**Model prediction :**

****

**Challenges Faced:**

* **Since we did not have access to high powered GPU, training time was very high which prevented us from experimenting with various model parameters.**
* **We were unable to run the model for longer iterations due to the processing issue.**
* **There were sudden spikes in accuracy due to multiple Brain images having no Tumor so nothing was detected by the model.**