Machine Learning: Course Final Project

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Title: Cascading Deep Learning Ensemble for Brain Tumor Segmentation from MRI Images

Team Members:

|  |  |
| --- | --- |
| **Name:**  MD Mahin  Anishkumar Chouhan  Bindiya Hassan Mohankumar | **PSID:**  1900421  1850356  1900778 |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## Index:

## Section 1: Survey

###### 1 (a): Paper No. 1: Brain Tumor Segmentation with Deep Neural Networks

###### 1(b) Paper No 2: Brain tumor segmentation with deep learning

###### 1(c)Paper No. 3: Classification using deep learning neural networks for brain tumors

###### 1(d) Paper No. 4: Automatic Brain Tumor Segmentation using Cascaded Anisotropic Convolutional Neural Networks

###### 1(e) Paper No. 5: U-net: Convolutional Networks for Biomedical Image Segmentation

## Section 2: Architecture: Cascading Deep Learning Ensemble

###### 2(a): The Proposed Ensemble

###### 2(b)All three-model’s configuration

## Section 3: Implementation and Performance

###### 3(a): Dataset Properties

###### 3(b): Implementation Details

###### 3(c): Performance

3(c)(i): Accuracy and Loss

3(c)(ii): Combined Performance and Comparison

## Section 4: Conclusion

## Section 5: References

## Section 1: Survey:

###### 1 (a): Paper No. 1: Brain Tumor Segmentation with Deep Neural Networks [1]

**Objective:** The main objective of this paper was to detect the location and extension of the tumor regions, namely active tumorous tissue (vascularized or not), necrotic tissue, and edema (swelling near the tumor). This was done by identifying abnormal areas when compared to normal tissue.

**Contribution:** This paper presents a fully automatic brain tumor segmentation method based on Deep Neural Networks (DNNs). A novel CNN architecture is presented which exploits both local features as well as more global contextual features simultaneously. [1]. The significant contributions of this work are: 1. proposed a fully automatic method with results. 2. Segmentation of a brain, this method takes between 25 seconds and 3 minutes. 3. The CNN implements a novel two-pathway architecture that learns about the local details of the brain as well as the larger context. Also proposed a two-phase training procedure which we have found is critical to deal with imbalanced label distributions. 4. Also employed a novel cascaded architecture as an efficient and conceptually clean alternative to popular structured output methods [1].

**Method:** They have used a task-adapted feature representations to learn a hierarchy of increasingly complex features directly from in-domain data. Since Deep neural networks have been shown to excel at learning such feature hierarchies [1]. This approach is applied to learn feature hierarchies adapted specifically to the task of brain tumor segmentation that combine information across MRI modalities. Since they have used images of brains in the BRATS dataset which lack resolution in the third dimension, segmentation performed slice by slice from the axial view. Thus, this model processes sequentially each 2D axial image (slice) where each pixel is associated with different image modalities namely; T1, T2, T1C and FLAIR. Like most CNN-based segmentation models, this method predicts the class of a pixel by processing the M X M patch centered on that pixel. The input X of this CNN model is thus an M X M 2D patch with several modalities [1]. The CNN here is a simple architecture corresponding to a single stack of several convolutional layers. The architecture use here is a two-pathway architecture. This architecture is made of two streams: a pathway with smaller 7X7receptive fields and another with larger 13 X 13 receptive fields. These streams are referred as the local pathway and the global pathway, respectively. This architectural method was chosen sine the prediction of the label of a pixel would be influenced by two aspects: the visual details of the region around that pixel and its larger “context”, i.e. roughly where the patch is in the brain.

**Performance:** The single path with one training phase CNN was ranked last with the lowest scores on almost every region. Using a second training phase gave a significant boost to that model with a rank that went from 15 to 9. The joint training of the local and global paths yields better performance compared to when each pathway is trained separately and the outputs are averaged.

###### 1(b) Paper No 2: Brain tumor segmentation with deep learning

**Objective:** This works main objective is to perform a pixel-wise classification. Learning on deep representations for each pixel based on its neighborhood in each modality (T1, T1c, T2 and Flair) and combine these to form a multimodal representation for each pixel.

**Contribution:** Segmenting brain tumors in multi-modal imaging data using Convolutional Neural Networks (CNNs) to perform the brain tumor segmentation task on the large dataset of brain tumor MR scans provided by BRATS2015 [2].

**Method:** To find tumors in brain images is to perform pixel-wise classification. An extraction of 32x32 patches in XY, YZ and XZ planes around each pixel for each modality is done. Where Deep Convolutional Neural Network (CNN) is used for each modality to learn good representations for every pixel based on the patches extracted surrounding that pixel. Then each CNN is trained separately to classify a pixel as one of non-tumor, necrosis, edema, non-enhancing, and enhancing [2]. Raw pixels from patches around each pixel form the input to the network. The softmax layer classifies the pixel as one of the five classes. Rectified linear unit is used in conjunction with the final hidden layer to improve gradients. The experiment was performed under two settings. In the first setting, a sample of random population of patches with equiprobable frequencies are used. The second setting makes use of all the patches from 20 randomly selected patients for training, and 5 for testing.

**Performance:** In the first setting, the training network consists of patches around 25000 randomly chosen pixels. Then the pixels are sampled so that their labels were in line with the distribution of the labels in the entire dataset [2]. The accuracy achieved was of 67% on a similarly sampled testing dataset. In the second setting the network was trained using all the patches of 10 patients and were able to reach a loss of 2.9 % on the training set.

###### 1(c)Paper No. 3: Classification using deep learning neural networks for brain tumors

**Objective:** The main objective in this work is to use a Deep Learning architectures for classifying a dataset of 66 brain MRIs into 4 classes e.g. normal, glioblastoma, sarcoma and metastatic bronchogenic carcinoma tumors [3].

**Contribution:** The contribution of this paper was to apply the deep learning concept to perform an automated brain tumors classification using brain MRI images and measure its performance. The main aim was to differentiate between normal brain and some types of brain tumors such as glioblastoma, sarcoma and metastatic bronchogenic carcinoma tumors using brain MRI images

**Method:** The proposed methodology for classifying the brain tumors in brain MRIs is as follows: 1. Brain MRIs Dataset acquisition 2. Image segmentation using Fuzzy C-means 3. Feature extraction using discrete wavelet transform (DWT) and reduction using Principle component analysis (PCA) technique 4. Classification using DNN. Basically in image segmentation, different normal brain tissues such as gray matter (GM), white matter (WM) and cerebrospinal fluid (CSF) and the skull from the tumor tissues in brain MR images [2] as the resulted segmented tumor part only would be used in the next steps. In Here Fuzzy C-means clustering technique is used to segment the image into 5 sections as it had good results in our previous work. After segmenting the Brain MR images into 5 sections features of the segmented tumor is extracted using discrete wavelet transform (DWT) [3].

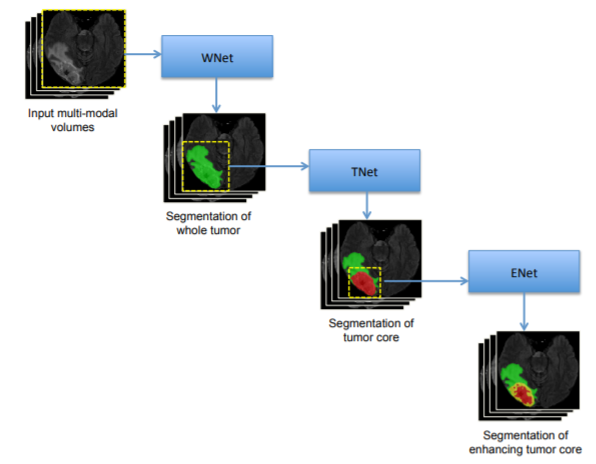
**Performance:** The performance of this work was measured in terms of average classification rate, average recall, average precision, average F-Measure and average area under the ROC curve (AUC) of all the four classes (normal, glioblastoma, sarcoma and metastatic bronchogenic carcinoma tumors) and compared to the performance of other classifiers in the same terms[3].

###### 1(d) Paper No. 4: Automatic Brain Tumor Segmentation using Cascaded Anisotropic Convolutional Neural Networks

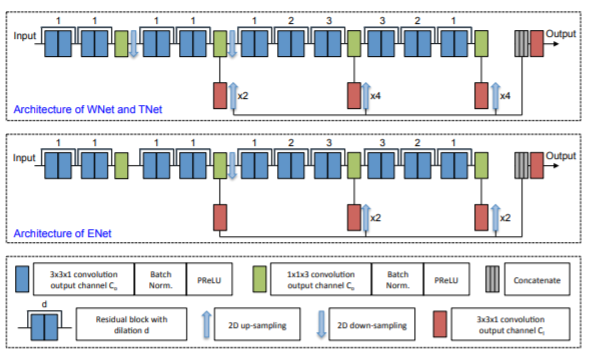
**Objective:** A cascade of fully convolutional neural systems is proposed to fragment multi-modal Magnetic Resonance (MR) images with brain tumor and three hierarchical levels: entirely tumor, tumor centre and enhancing tumor centre.

**METHODS:**

1. **Triple Cascade Framework**: The fig 1 shows the proposed cascade framework. There are three networks which are hierarchically and sequentially fragmented substructure of brain tumor, and each of these networks deals with a binary segmentation of problem. The entire tumor from multi-modal 3D volumes of the same patient is fragmented in the first network (WNet)and is obtained by the bounding box. Based on the bounding box the input images are cropped and are used as the input to the second network(TNet) to fragment the tumor centre. The Input to the third network (ENet) are the bounding box of the tumor centre which are fragmented to enhance the tumor core in the third network. The bounding box are automatically generated based on the ground truth in the training stage. [4]



**Fig 1.** The proposed triple cascaded framework for brain tumor segmentation



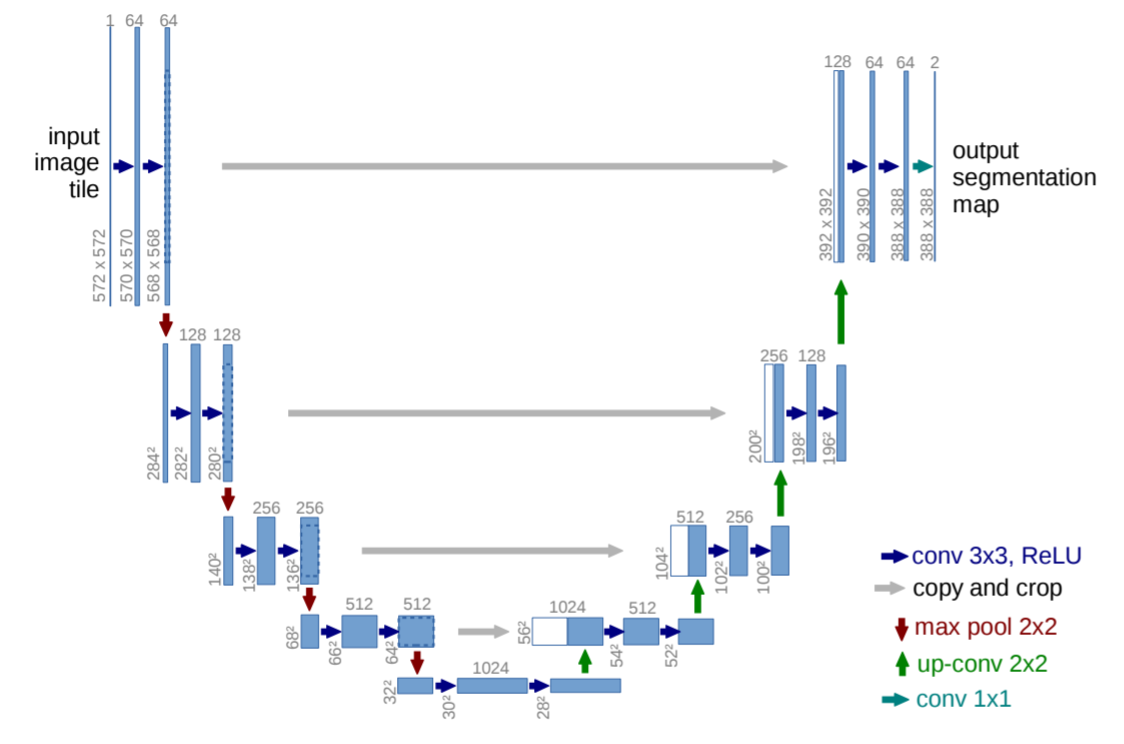
**Fig 2.** Anisotropic convolutional networks with dilated convolution, residual connection and multiscale fusion.

1. **Anisotropic Convolutional Neural Networks:** Fig2 shows the proposed architecture of the network. A large receptive field in 2D and a relatively small receptive field in the out-plane direction i.e., orthogonal to the 2D slices are taken as a stack of slices as input in the anisotropic. The WNet, TNet and ENet are 217x217, 217x217 and 113x113 respectively are 2D receptive field. Typically, the 2D sizes of the inputs are smaller than the corresponding 2D receptive field during training and testing.
2. **Residual Kernel:** WNet, TNet and ENet have 10 residual blocks. Each of these blocks contains two intra-slice convolution layers, and the input of a residual block is directly added to the output. This can make information propagation smooth and speed the convergence of training.
3. **Multi-Scale Prediction:** To get multiple intermediate prediction three 3x3x1 convolution layers at different depths of the network are used and upsample them to the resolution of the input, so that the features are combined. A prediction of these concatenation is fed into an additional 3x3x1 convolution layer to obtain the final score map. In fig 2 the red block shows these layers.
4. **Multi-view Fusion:** The fragmentation results from three different orthogonal views are fused. WNet, TNet and ENet are trained axial, sagittal and coronal views respectively. The predictions of these three views during testing time are fused to get the final segmentation. The softmax output are averaged in these three views for each level of the cascade.
5. **Anisotropic and Dilated Convolution**: A 3D kernel is fragmented with a size of 3x3x3 into an intra-slice kernel with a size of 3x3x1 and an inter-sliced kernel with a size of 1x1x3. The kernel has *Co* output channels and followed by a batch normalization layer and an activation layer (Parametric Rectified Linear Unit PReLU) as shown in blue and green blocks in fig 2. WNet and TNet use 20 intra-slice convolution layers and four inter-slice convolution layers with two 2D downsampling layers. Considering the smaller input size ENet uses only one downsampling layer and same set of convolution layer as WNet. After downsampling layers, the dilated convolution used to enlarge the receptive field within a slice in intra-slice kernel. The parameters are as shown in fig 2.

###### 1(e) Paper No. 5: U-net: Convolutional Networks for Biomedical Image Segmentation

**Objective:** In this paper, authors presented a network and training strategy based on the heavy use of data growth in order to make more efficient use of accessible annotated samples.

**METHOD:**



**Fig 1.** U-net architecture (example for 32x32pixels in the lowest resolution).

The network architecture is as shown in Figure 1. On the left side of the architecture it has a contracting path and On the right side an expansive path. The architecture of the contracting path consists of a convolutional network. The downsampling of the U-net of the repeated application of two 3×3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2×2 max pooling operation with stride 2. The feature channels are doubled at each downsampling step. Each step in the expansive path consists of an upsampling of the feature map followed by a 2×2 convolution (up-convolution) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3×3 convolutions, then followed by a ReLU. There is border pixel loss in each convolution, hence cropping is needed. The 64-component feature vector to the desired number of classes is obtained by convolution of a 1x1 in the final layer.The overall network has 23 convolutional layers**.** [5]

## Section 2: Architecture: Cascading Deep Learning Ensemble

In this project we aimed to segment brain tumors using deep learning. For our project we have created a cascading ensemble consisting of three different model. All the models are improvement of UNet architecture. All the models have one down-sample block, one bottleneck and one up-sample block. However, within the blocks they are configured differently. We have taken the models that performed very well for our dataset. The input image size is used 512\*512, as we have seen with this shape the performance is better. The configuration of three architecture is given below:

###### 2(a): The Proposed Ensemble:

Our proposed model is sequentially trained ensemble. As we have used separate models to train sequentially, it is like cascading. The architecture of the model is built using the following procedure:

Model 1

Model 2

Model 3

All Input

Samples with loss more than threshold from Model 1

Samples with loss more than threshold from Model 2

Training

Training Steps:

* 1. Set: batch size, max epochs, loss threshold
  2. Set train batch = train data, validation data = validation data, test data = test data
  3. Set model = first model
  4. Train the model batch by batch
  5. If epoch == max epoch:
     1. Check if batch loss > loss threshold:
        1. Add batch id to new\_batch
  6. Set train data = new train data
  7. Set model = next model
  8. Set new epoch size
  9. If last model trained
     1. End
  10. Else: go to step Iv

Model 1

Model 2

Model 3

Testing

Sample

Output with more boolean True

Output

Testing Steps:

1. For sample in training sample:
   1. Predict the output y1, y2, y3 using model 1, model 2 and model3
   2. Choose output with greatest sum

Here, for the training and testing steps we have a number of suggestions:

1. Max epoch needed to be number of times till average loss has reached till the certain threshold or the model from the ensemble starts overfitting or a highest number reached. We did not add this in our procedure because we could not train each model till they overfit.
2. For testing or output, more sophisticated procedure can be designed. At present we have used only the max output as max output will have maximum 1’s. However, there any type of confidence can be used and using that confidence the output can be averaged. We did not add that to our model as we could not test that.

###### 2(b)All three-model’s configuration:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model 0 | | | Model 2 | | | Model 3 | | |
| Symbol | Layer | Configuration | Symbol | Layer | Configuration | Symbol | Layer | Configuration |
|  | Input | 512 \* 512 \* 3 |  | Input | 512 \* 512 \* 3 |  | Input | 512 \* 512 \* 3 |
| C1 | Convolution | 8@3\*3, ReLU | C1 | Convolution | 8@5\*5, ReLU | C1 | Convolution | 8@3\*3, ReLU |
|  | Batch Normalization |  |  | Batch Normalization |  |  | Batch Normalization |  |
|  | Dropout | 10 to 25 % |  | Dropout | 10 to 25 % |  | Dropout | 10 to 25 % |
| P1 | Pooling | 2\*2 | P1 | Pooling | 2\*2 | C2 | Convolution | 8@5\*5, ReLU |
| C2 | Convolution | 64@3\*3, ReLU | C2 | Convolution | 64@7\*7, ReLU |  | Batch Normalization |  |
|  | Batch Normalization |  |  | Batch Normalization |  |  | Dropout | 10 to 25 % |
|  | Dropout | 10 to 25 % |  | Dropout | 10 to 25 % | P1 | Pooling | 2\*2 |
| C3 | Convolution | 64@3\*3, ReLU | C3 | Convolution | 64@7\*7, ReLU | C3 | Convolution | 64@7\*7, ReLU |
|  | Batch Normalization |  |  | Batch Normalization |  |  | Batch Normalization |  |
|  | Dropout | 10 to 25 % |  | Dropout | 10 to 25 % |  | Dropout | 10 to 25 % |
|  | Upsampling | 2\*2 |  | Upsampling | 2\*2 | C4 | Convolution | 64@7\*7, ReLU |
| C4 | Convolution | 8@3\*3, ReLU, Cascaded with C1 | C4 | Convolution | 8@3\*3, ReLU, Cascaded with C1 |  | Batch Normalization |  |
|  | Batch Normalization |  |  | Batch Normalization |  |  | Dropout | 10 to 25 % |
|  | Dropout | 10 to 25 % |  | Dropout | 10 to 25 % |  | Upsampling | 2\*2 |
| C5 | Convolution | 1@3\*3, sigmod | C5 | Convolution | 1@3\*3, sigmod | C5 | Convolution | 8@3\*3, ReLU, Cascaded with C2 |
|  | Output | 512 \* 512 \* 1 |  | Output | 512 \* 512 \* 1 |  | Batch Normalization |  |
|  |  |  |  |  |  |  | Dropout | 10 to 25 % |
|  |  |  |  |  |  |  | Convolution | 8@3\*3, ReLU, Cascaded with C1 |
|  |  |  |  |  |  |  | Batch Normalization |  |
|  |  |  |  |  |  |  | Dropout | 10 to 25 % |
|  |  |  |  |  |  | C7 | Convolution | 1@3\*3, sigmod |
|  |  |  |  |  |  |  | Output | 512 \* 512 \* 1 |

## Section 3: Implementation and Performance:

###### 3(a): Dataset Properties:

We have collected our data from Kaggle brain tumor dataset. The dataset contains of mri images with brain tumor and the mask. There are a total of 1375 samples from 110 patients with their corresponding masks. All the sample data has tumors.

###### 3(b): Implementation Details:

We have implemented our model on google colab platform. Our dataset is divided into 70% for training, 15% for testing and 15% for validation. We have trained each model for 50 epochs. Callback function is designed to track the performance for each training sample.

###### 3(c): Performance:

We have trained our model sequentially. First, we have trained a model, then based on the performance of the model we have selected a portion of the dataset and trained the next model. This is how our model performed:

###### 3(c)(i): Accuracy and Loss:

First model is trained with 963 samples. The lass and accuracy from each epoch are plotted in the graph. From the figure 1 we can see that, our validation loss fluctuate in between 13 to 14 at the end of the epochs but overall training loss constantly decreasing and ends in between 11 to 12. For accuracy from figure 2, we can see that training and validation accuracy settles down at the end of the epochs in between 96.5% to 97%. This model has almost 50% of its sample with loss below 10%. Second model is trained with 500 samples for which previous models had lost more than 10%. Accuracy and loss pattern are more stable for this model, but the final result ends up within the same domain as model 1, fig 3 &4. Third model is trained with 277 samples for which previous models had loss more than 10%. ends up within the same domain as model 1 and 2 here, fig 4,5. From the graphs it is apparent that, none of the model overfitted.

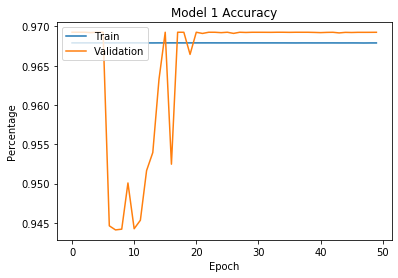
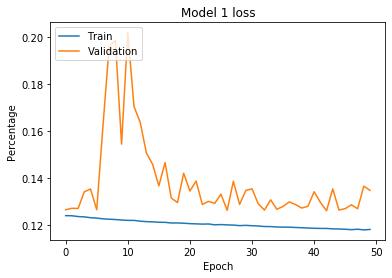


Fig 1: Model 1 train vs testing loss Fig 2: Model 1 train vs testing accuracy

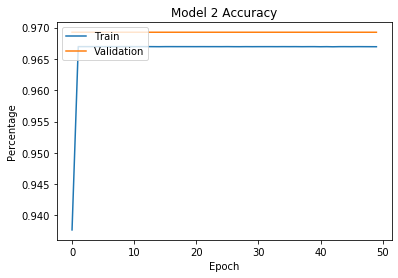
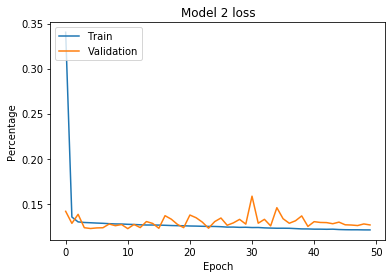


Fig 3: Model 2 train vs testing loss Fig 4: Model 2 train vs testing accuracy

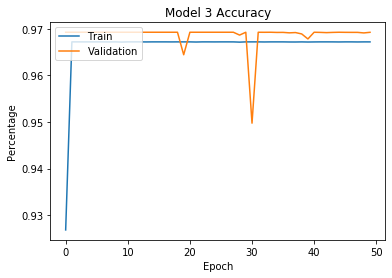
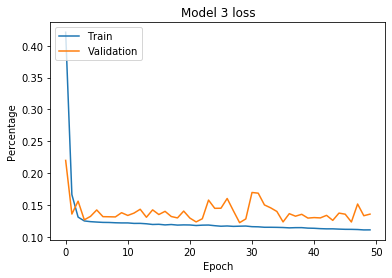


Fig 5: Model 3 train vs testing loss Fig 6: Model 3 train vs testing accuracy

###### 3(c)(ii): Combined Performance and Comparison:

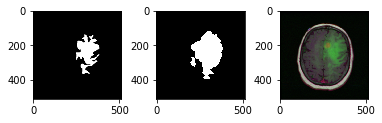
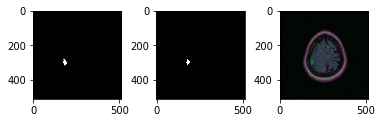


Fig 7,8: Segmentation of our model (Mask + Result + image)

For segmentation our model performed better than UNet. Fig 7,8 shows some segmentation done by our model. In comparison segmentation done by UNet given below.

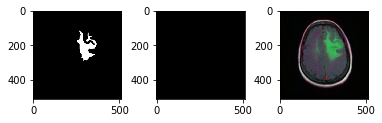
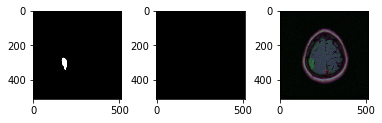
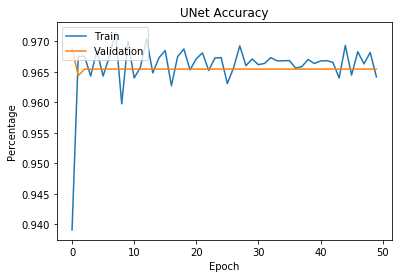
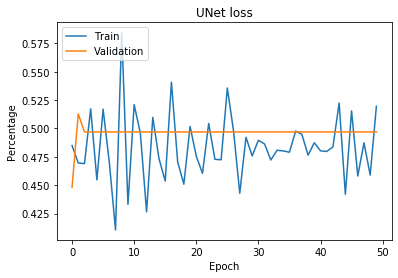


Fig 9,10: Segmentation of our model (Mask + Result + image)

It is visible that our model is performing better. For loss and accuracy our model is performing better than the UNet. The output from the UNet is quite unstable with loss much higher than us.



**Fig 11,12**: UNet Loss and Accuracy

## Section 4: Conclusion:

The aim of this project is to segment brain tumour from mri images using deep learning. For this purpose we have taken UNet as our baseline architecture. However the exact UNet architecture performs poorly for our dataset. That is why we modified the baseline UNet in three variations. Later we have created a cascading ensemble using the three model. Our approach of developing the ensemble is: first we have trained a model and based on the sample loss from the model we have trained the next model, only with samples having loss above a certain threshold. The reason is a model does not perform well for all samples. This way we have created multiple experts for different types of sample. Later, the output is the output with max boolean true, as they have more one as value. Our limitation is, due to our hardware limitation we could not verify exact accuracy of the whole model and could not test the model in ultimate cases where each version overfits. We also could not compare it with highest configuration of UNet model.

## Section 5: References:

[1] Brain Tumor Segmentation with Deep Neural Networks, Mohammad Havaeia, Axel Davyb, David Warde-Farleyc, Antoine Biardc, Aaron Courvillec, Yoshua Bengioc, Chris Palc, Pierre-Marc Jodoina, Hugo Larochellea.

[2] Brain Tumor Segmentation with Deep Learning Vinay Rao , Mona Sharifi Sarabi , Ayush Jaiswal

[3] Classification using deep learning neural networks for brain tumors Heba Mohsen, El-Sayed, El-Dahshan, El-Sayed M. El-Horbaty, Abdel-Badeeh M. Salem

[4] Guotai Wang, Wengi Li, Sebastien Ourselin and Tom Vercauteren: Automatic Brain Tumor Segmentation using Cascaded Anisotropic Convolutional Neural Networks

[5] Olaf Ronneberger, Philipp Fischer, and Thomas Brox.: U-net: Convolutional Networks for Biomedical Image Segmentation.