An Active-Learning based Training-Schedule for Biomedical Image Segmentation on Deep Neural Networks

by

Mehadi Hassan
17101177
Shemonto Das
17101447
Shoaib Ahmed Dipu
17101482

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
January 2021

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. The thesis acknowledges all main sources of help.

Students' Full Name & Signature:

Mehadi Hassan
17101177

Shemonto Das
17101447

Shoaib Ahmed Dipu
17101482

Approval

The thesis titled "An Active-Learning based Training-Schedule for Biomedical Image Segmentation on Deep Neural Networks" submitted by

1. Mehadi Hassan (17101177)
2. Shemonto Das (17101447)
3. Shoaib Ahmed Dipu (17101482)

of Fall, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on January 08, 2021.

Examining Committee:

Supervisor:
(Member)

Mahbubul Alam Majumdar, PhD
Professor and Dean
School of Data and Sciences
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Sowmitra Das
Lecturer
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Shahnewaz Ahmed
Lecturer
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Mahbubul Alam Majumdar, PhD
Professor and Dean
School of Data and Sciences
Department of Computer Science and Engineering
Brac University

Abstract

Biomedical image classification and segmentation are quite important tasks for medical diagnosis. Many Deep Neural Networks (U-Net, V-Net, etc) have been used in recent years to segment biomedical images. For classification of biomedical images or 3D data (X-Ray, CT scan, MRI), ResNet, DenseNet, Xception, Inception, etc. have been in use for automatic disease diagnosis. But all of these networks are trained end-to-end and they do not accumulate anatomical information that is required to interpret similar data in the same way Radiologists do. A new research direction would be to make the network aware of key anatomical locations and their relative positions while generating predictions. We investigated the roles that Active Learning can play in the development and deployment of Deep Learning enabled diagnostic applications and focus on techniques that will retain significant input from a human end-user. In order to practically understand the drawbacks of existing approaches using different networks, we benchmarked the MICCAI BraTS 2019 dataset on different Neural Networks. To overcome the drawbacks of existing approaches of different networks we have incorporated an uncertainty-based Active Learning Training Schedule to segment biomedical images. Through this approach, we have achieved a much better performance than the traditional end-to-end approaches on Deep Neural Networks for biomedical image segmentation.

Keywords: Active Learning; Deep Learning; Uncertainty Metric; Biomedical Image Segmentation

Dedication

Dedicated to all the Computer Vision practitioners who are relentlessly working in the medical domain and constantly trying to improve it.

Acknowledgment

Firstly, all praise to the Almighty for blessing us with sound health during this pandemic for which we have been able to complete our tasks in due time.

Secondly, gratitude to our supervisors - Mahbubul Alam Majumdar, Sowmitra Das and Shahnewaz Ahmed for their relentless efforts to guide our research works.

Thirdly, we would like to thank Asif Shahriyar Sushmit (Research Assistant, mHealth Lab, BME, BUET) who suggested us the dataset on which we worked and use Active Learning or Human-in-the-Loop Computing for Biomedical Image segmentation.

Fourthly, we would like to thank Saddat Hasan for providing us technical support and Ahmed Rakin Kamal for providing us logistic support when required.

And, finally, we want to express our gratitude to our families. Without their continuous support, it would not be possible to pursue this research work as well as this Undergraduate Degree.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AL Active Learning

$BraTS$ Brain Tumor Segmentation

CNN Convolutional Neural Network

DL Deep Learning

HGG High Grade Glioma

$HITL$ Human-in-the-loop

LGG Lower Grade Glioma

ML Machine Learning

$ReLU$ Rectified Linear Unit

Chapter 1

Introduction

1.1 Introduction

Biomedical images like MRI, X-ray, CT Scan are widely used to determine a patient's diseases by providing a broader look into the internal organs of the human body. Segmenting such an image is the main concern in Biomedical Image Processing and it is also considered as the first step of analysis procedures[1], [3]. The segmented images are partitioned into multiple regions based on the interest. In practical life, these images are manually checked by professionals and are used to provide treatment. So, automating these services provides an improvement in diagnostic confidence and accuracy[33].

From biomedical images, it is difficult to identify something due to noise and modality-specific artifacts. It requires experts having prior knowledge to provide a decision. As a result, this does not directly work from one clinical model to others due to domain gap[25] and requires constant data annotation by the annotators. Due to improvement in computer hardware and computer vision different Deep Learning models have been implemented which can achieve same accuracy as a human observer. However, the use of these Deep Learning models is limited because of complex datasets, unique challenges and trust issues in the trained models.

Budd et al. [36] identified three key challenges when developing Deep Learning based application for medical image analysis in a clinical setting which are, Lack of Training Data, Improving the Final Percentage and Interpreting the Predictions. Transparency and Interpretability though will be our biggest concern while we discuss more about semantic segmentation of the medical images. Existing deep learning models primarily concentrate on the development of predictive models for a specific task and demonstrate the state-of-the-art-performance for that task.

We would focus on medical image segmentation techniques where humans will play the roles of end-users and also explore their roles in Deep Learning (DL) enabled systems. This concept leads us to the introduction of 'Active Learning based Training Schedule'. That is we will move away from the traditional end-to-end approach and develop an efficient, robust and optimized Active Learning based Technique that will result in a higher true accuracy level than the existing models.

We have divided our tasks into three separate parts. Firstly, to grasp the concepts of both existing and Active Learning based approaches. Secondly, to reproduce the current benchmark-results of the existing models. Finally, to implement our proposed Active Learning based Training Schedule to solve the drawbacks of the existing models for biomedical image segmentation. We would mainly work on MICCAI BraTS 2019 dataset to conduct our research works.

1.2 Problem Statement

Over the years due to improvements in computer hardware and Deep Learning (DL) algorithms, these techniques have been providing very good accuracy. Despite having good accuracy, these techniques are still less used in medical image segmentation as we are still not sure how well this predictive model will perform in real-life scenarios. As we are talking about patients, accuracy matters a lot over here. These DL methods have achieved state of the art performance but if we consider safety measurement, even the silliest mistake can lead to a catastrophic disaster. According to Budd et al. [36] these DL methods are considered as 'Black Box' because end users have a very limited way of interacting with the models. In most cases, we feed the network with data and after doing calculations the network provides an output. Budd et al. [36] also mentioned that in Biomedical image segmentation transparency is important. To make a clinical decision it needs to be verified from different sources to minimize the error near to zero. But existing DL methods do not provide such features like human interaction. In order to rely on the network, we must allow users to weigh automated predictions. Furthermore, current networks do not have any idea where to look specifically in the image. As a result, models iterate over the pixels and try to identify a pattern to make a decision based on it and which is not efficient to predict the correct outcome and in addition to that lot of computational power is wasted as well. To overcome these problems we need to introduce Active Learning which will allow us to provide more interaction with the network.

1.3 Research Objectives

The main goal of our research will be to improve the traditional Deep Learning (DL) based techniques by introducing the concept of user-end input or oracle intervention in the network which is known as 'Active Learning'. To elaborate, we want to see through the 'Black Box' in DL. As of now, only the network input has been the only means of communicating with the networks and the rest was left on the network thinking the network itself will somehow (which has been the 'Black Box') provide output with higher accuracy and perfection. However, in practice, it is not always possible for a network to be that perfect and any error even if the slightest in medical image segmentation can question the whole diagnosis process. As a solution to it, we proposed user end input along with the traditional input to the network as an Active Learning based Training Schedule approach. For this purpose, we would like to introduce uncertainty as our metric that will help us to determine which particular data to work on while the model is being trained. We strongly believe that the intervention from the user during the training procedure will help the network shape towards a legitimate output. Moreover, the directed supervision will also result in higher confidence output.

1.4 Research Workflow

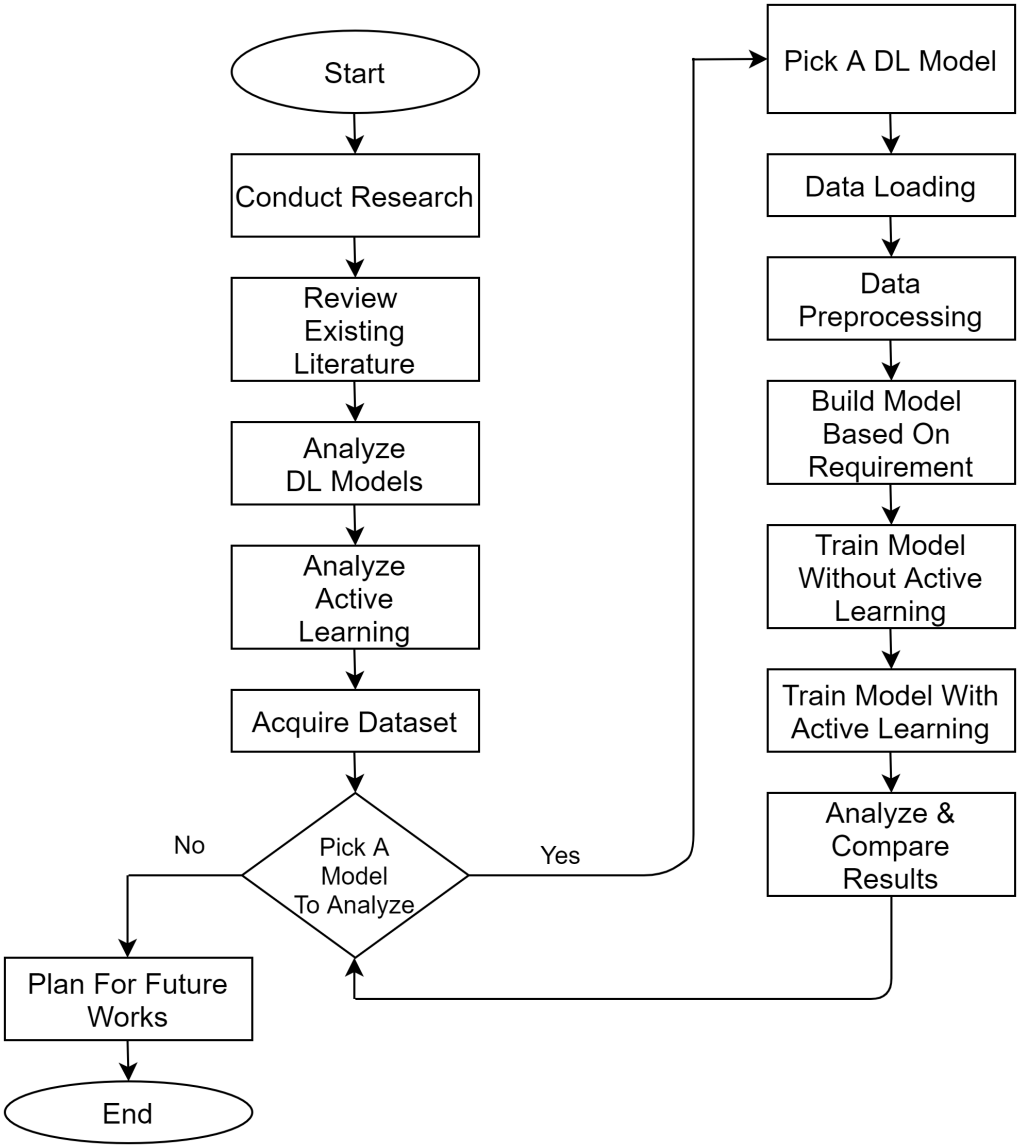

Figure 1.1: Workflow Diagram

Chapter 2

Literature Review and Relevant Work

2.1 Deep Convolutional Neural Networks

Deep Learning (DL) techniques such as Convolutional Neural Network (CNN) and variants of this are in consideration as the cutting edge techniques to date for semantic image segmentation and have been applied to the processing of medical images. Long et al. [10] provided a fully convolutional network which will label pixel-wise consisting of Convolution, ReLU, Pooling, and Fully Connected Layer. Primarily for detecting an object from images, it is necessary to classify the appropriate regions and apply CNN to them. One of their proposal to calculate pixel-wise output is deconvolving the output of activation maps. Another proposal of theirs was to make a combination of shallow layers output with the output of the network which will help in extracting the in-depth details of the multidimensional medical images. But, to incorporate Machine Learning(ML) in medical images, some constraints are there due to incompleteness and shortage of labeled datasets. The low number of labeled datasets leads to having an overfitting issue. Despite the high performance of CNN based methods on medical images, the constraints of training with a low number of labeled datasets are still there [2][17]. To achieve exact localization Ronneberger et al.[12] proposed an architecture including a contracting path that will capture context and an expanding path which have symmetry. The architecture is known as U-Net. With the emergence of SegNet which is an encoder-decoder based segmentation network, to classify pixel-wise, low-resolution encoder feature can be mapped to full input resolution feature maps [19][11]. Taghanaki et al.[39] mentioned that SegNet has a new approach in the way of upsampling lower resolution input feature maps with the help of its decoder. This network has been built with multiple convolutional layers having batch normalization, with an activation function ReLU and later on max-pooling which doesn't overlap and subsampling. Such architecture has provided the network an upper hand in classifying images at a pixel level while preserving very finite details about the localization of modality within the medical images. This type of structure has enabled the network to classify images at pixel level saving the finite details about localization of modality into medical images. When training is done from scratch, precise initialization of weight can improve gradient vanishing issue [26]. Here, the weight of the kernel has been initialized with the help of sampling from a distribution known as the normal distribution. Milletari

et al.[16] proposed a network which is derived from U-Net and known as V-Net that brought in the inclusion of convolutions in contracting path to extract and detect feature more accurately. Unlike U-Net which is applicable for 2D data, this network can work on 3D data. Convolution window along with size of kernel and stride has been taken into great consideration which allows the network to have a much wider map for segmentation. The main aim of all the tasks related to biomedical image segmentation was to have a result that can be used in a clinical setting. But the continuous limitations in this field is the complexity of the biomedical images and their modality. Alongside this, maintaining the clinical precision for 4D, 5D images become more difficult with these end to end segmentation models. A good number of encoder-decoder based networks has been altered by making them deeper or shallower or by including more attention blocks than required for semantic segmentation purpose [32][35] and quite a few of them performed much better than many cutting edge networks for segmentation. But, all of these networks didn't include any intervention of user by being end to end based networks.

2.2 U-Net

Ronneberger et al. [12] proposed an architecture including a contracting path that will capture context and an expanding path which have symmetry to achieve exact localization. They also added a skip connection which resulted in enhanced accuracy. Moreover, they also addressed the problem of vanishing gradients. This model is known as the U-Net which has played a significant role in biomedical image segmentation and included the concept of deconvolution introduced by Zeiler et al. [9] and is built upon the elegant architecture of Fully Convolutional Network.

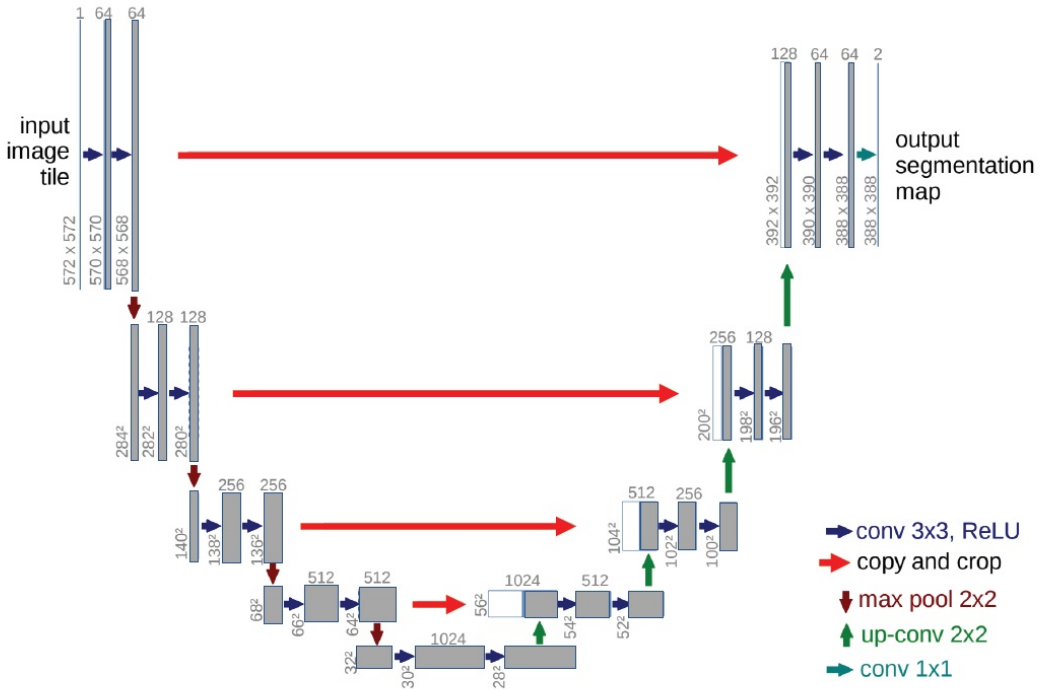

Figure 2.1: An illustration of the U-Net [12] architecture.

Hesamian et al. [37] mentioned the benefit U-Net received from skip connections between different stages of the network and brought about some changes to deal with the trade-off between localization and the use of context. More pooling layers are required to handle patches of large size which results in the reduction of localization accuracy. Alongside this, Hesamian et al. [37] accentuated more on the shortcut connections of U-Net between the layers of same resolution in analysis path to expansion path, and for this reason, these connections provide necessary high-resolution features to the deconvolution layers. It is highly recognized in biomedical image segmentation and many other networks have been built on this architecture. [13], [29], [31]

2.3 Active Learning

Lack of annotated training data has been one of the challenges for supervised Deep Learning techniques as they depend on large and properly distributed and accurately annotated data points. Even though more medical image datasets are becoming available, the time, cost and effort which is needed to annotate such datasets remain quite effortful. To address this issue, Active Learning has emerged. As Budd et al. [36] mentioned, Active Learning (AL) tries to find this optimal subset L^* given a current model $f'(x|L')$, where L' is an intermediate annotated dataset, and an unannotated dataset U. Active Learning methods try to find out the most informative data points X_i^* to train a model assuming that both the model and the un-annotated dataset will evolve with the course of time instead of selecting a fixed subset once for training. Budd et al. [36] also mentioned that, as soon as new annotations would be acquired, the AL framework will have to use the new data to bring an improvement to the model. This can be done by retraining the entire model using the annotated data L' which are available. Apart from retraining, fine-tuning can also be done using the annotated data-points X_i^* to improve the model.

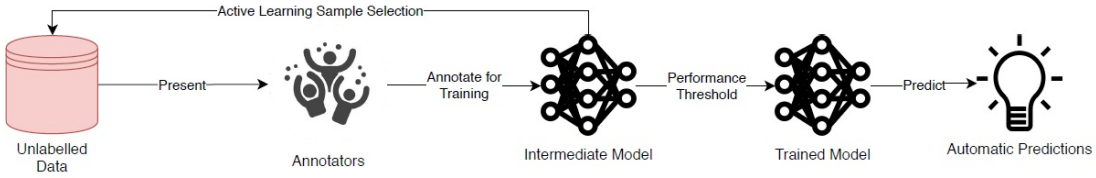

Figure 2.2: Overview of Active Learning frameworks [36]

2.3.1 Interpretability and Refinement

For the development of the Active Learning(AL) framework, the informativeness of the data needs to be measured for proper segmentation of biomedical images. According to Budd et al. [36], hand-based heuristics are used for AL methods to enhance the informativeness of the data. This enhanced informativeness will be an increasing factor for the detection of minute regions of the modalities in biomedical images. Budd et al. [36] argue that more information can be gained with increased uncertainty of prediction by making the ground truth for that sample a part of the dataset. A prominent strategy to find the most 'valuable' areas is to use uncertainty sampling, with the active learner querying the most uncertain areas for directed training. Budd et al. [36] mentioned entropy as the most popular approach for uncertainty measurement as it generalizes well as an uncertainty measure. Konyushkova et al. [38] came up with an AL-based approach that uses entropy as uncertainty metrics for the estimation of a pixel that will be annotated next This entropy-based uncertainty metric is the calculation of pixel-wise entropy based on each pixel probability which is a result of binary classification. For each pixel probability, the whole probability distribution needs to be taken into account. The purpose of the whole uncertainty-based AL framework is to correct the probable error of each pixel which is indeed the goal of semantic segmentation. Finally, the overall uncertainty of each training sample is computed as the mean uncertainty of

its pixels. According to Yang et al. [28], at the end of each stage of Active learning-based training, data with the highest entropy that is uncertainties are extracted and those are decided to be the next batch of data to be trained. In the traditional training approach, we blindly fit the data into the model vesting the whole training responsibility upon the 'Black Box' which restricts us from the human-based heuristic refinement causing the whole deep learning-based modality detection a clinical drawback.

2.3.2 Uncertainty

Informativeness measure is mainly calculating uncertainty. It has been said that we can have more information from a more uncertain prediction than a somewhat less certain prediction than the previous one. For this case, we need to include ground truth for those samples in the training set.

There are multiple ways to calculate uncertainty from many ML/DL models.

Least Confident : In this way, the sum of the lowest class probability is calculated. It is known that those who have more certain predictions, highly likely there will have higher pixel-wise class probabilities. This approach has got a major drawback which is discarding the information of the rest of the label distribution.

Margin Sampling : This approach [4] solves the lacking of Least Confident as it takes the first and second most probable labels under consideration and computes the difference between them. The more the difference the more confident the model is.

Entropy : If we consider Binary Classification, this approach is as same as Least Confident and Margin Sampling. But for Multi-Class problems, Entropy acts as a quite good approach as an uncertainty measure.

Complimentary Sampling : Wang et al. [18] provided another approach in which two types of selections are done. Firstly, sets of uncertain samples are selected to be labeled by oracles. Secondly, sets of higher certain samples are pseudo-labeled before getting included in the labeled dataset.

Bayesian CNN : Gal et al. [24] proposed that Bayesian CNN performs much better than deterministic CNNs in this arena of Active Learning. In this approach, multiple types of queries are incorporated.

2.4 Challenges of Training Deep Learning Models

2.4.1 Overfitting

Overfitting occurs when captured patterns and regularities by a model in the training set are unusually more accurate with a comparison to the unprocessed instances of the problem [14]. If the size of the dataset is small, overfitting occurs. By bringing an increment in the size of the dataset, it is assumed that the issue will be solved [27]. For example, data augmentation has been proved to be an effective way of handling overfitting issue [23]. Alongside this, we can get rid of overfitting by dropping sets of neurons during the training [8]. A modified approach of dropping neurons is dropping connections which also solves this issue [6].

2.4.2 Training Time

Achieving a training time that is less than usual and to have a convergence quickly have been in the study in many cases. By applying pooling layers dimensionality of the parameters can be reduced and which will eventually solve this issue [22]. Recently proposed pooling based solutions use convolution with stride [7] which makes the network less heavy than usual. Batch normalization performs better to bring an improvement of achieving quicker convergence [13], [15], [21] as it does not hamper performance like pooling and down-sampling which may let us loosing necessary information.

2.4.3 Gradient Vanishing

Deeper networks usually perform better despite their struggles of exploding or completely vanishing of the propagated signal which is known as gradient [26]. To be more specific, it is quite difficult to backpropagate the final loss to shallow layers. 3D models suffer from this issue quite often. By scaling up intermediate hidden layers' output using deconvolution and passing it to a softmax to get the prediction can solve this issue. Scaled up the output of immediate hidden layers with the help of deconvolution operation and using this output as an input for activation function Softmax can solve this issue. Combining auxiliary loss with original loss of the hidden layer can strengthen the gradient [22], [29], [34]. If we cautiously initialize weight while doing scratch training, it can bring an improvement to solve this issue of training [26].

2.5 Challenges of Training Deep Learning Models in Clinical Setting

Budd et al. [36] mentioned about three key challenges of medical image analysis in the clinical setting:

2.5.1 Lack of Training Data

Regular supervised Deep Learning techniques mostly depends on adequately large datasets that are accurately annotated. But, there are not adequate datasets for Medical Images and annotating them is quite time and cost consuming. It requires great effort too.

2.5.2 The Final Percent

Even though Deep Learning techniques have achieved the best possible performance till date for Biomedical Image Classification and Segmentation, in critical conditions any deviation or misinterpretation can lead to destruction. If we want to achieve a credible result, we need to incorporate interactive interpretation by including an oracle in the loop.

2.5.3 Transparency and Interpretability

Most of the Deep Learning models are considered as the 'Black Box' as we can hardly have the scope of understanding the underlying mechanisms of decision mechanism of a particular model. To rely on the result, we need to have the scope of understanding the decision-making process, or no matter how good a model provides an output, we will have a hard time relying on that.

2.6 Metrics for Evaluating Segmentation Models

Evaluation of the segmentation model is performed using different metrics. Some are,

2.6.1 Pixel Accuracy

It denotes the ratio of pixels properly classified, divided by the total number of pixels. Pixel accuracy can be defined as,

$$\text{PA} = \frac{\sum_{i=0}^{K} p_{ii}}{\sum_{i=0}^{K} \sum_{j=0}^{K} p_{ij}}$$

K = foreground classes and background
p_{ij} = number of pixels of class i predicted as belonging to class j

2.6.2 Mean Pixel Accuracy

It is an enhancement of Pixel Accuracy. Here the ratio of correct pixels is calculated in a per class manner. After that, it is averaged over the total number of classes.

$$\text{MPA} = \frac{1}{K+1} \sum_{i=0}^{K} \frac{p_{ii}}{\sum_{j=0}^{K} p_{ij}}$$

2.6.3 Intersection over Union (IoU)

At first, the area of intersection between the predicted segmentation map and ground truth is calculated. Then, the area of union between the predicted segmentation map and ground truth is calculated. Finally, the area of intersection is divided by the area of union. It ranges between 0 and 1.

$$\text{IoU} = J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

A = Ground truth, B= Predicted segmentation maps

2.6.4 Mean-IoU

It is calculated by taking the average Intersection over Union (IoU) of all classes. This metric is mostly used for reporting the performance of modern segmentation algorithms.

2.6.5 Precision / Recall / F1 Score

These metrics are used to evaluate the accuracy of many image segmentation models. We can define Precision and Recall both for each class and aggregate level.

$$\text{Precision } = \frac{\text{TP}}{\text{TP} + \text{FP}}, \text{ Recall } = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{TP} = \text{True Positive, FP} = \text{False Positive, FN} = \text{False Negative}$$

Another metric, F1 Score is a combination of Precision and Recall. It is the harmonic mean of them.

$$\text{F1-score } = 2 \cdot \frac{\text{Precision } \cdot \text{ Recall}}{\text{Precision } + \text{ Recall}}$$

2.6.6 Dice Coefficient

The Dice coefficient, or Sørensen–Dice coefficient, is a common metric for pixel segmentation as it can also be modified to act as a loss function. We can multiply the overlapped area of two images by 2 and then need to divide it by the value of the total number of pixels from both images.

$$\text{Dice } = \frac{2|A \cap B|}{|A| + |B|}$$

This original formula of Sørensen was applicable for discrete data. It was later modified to be applied for Boolean data and then it became identical to F1 Score.

$$\text{Dice } = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} = \text{F1}$$

Dice Coefficient can be defined as vector operations too.

$$\text{Dice} = \frac{2|\mathbf{a} \cdot \mathbf{b}|}{|\mathbf{a}|^2 + |\mathbf{b}|^2}$$

Chapter 3

Dataset Handling, Implementation and Research Methodology

3.1 Dataset

We acquired the dataset from a challenge entitled 'Multimodal Brain Tumor Segmentation Challenge 2019 (BraTS 2019)' organized by Medical Image Computing and Computer-Assisted Intervention (MICCAI) Society. Apart from serving the purpose of this competition, this is also available for usage of research [5] [20] [30].

To build this dataset, MRI tumor scans from multiple test centers were collected as per the standard clinical conditions. Dataset is formatted as NIfTI files (.nii.gz) and categorized as T1, T2, Flair and T1Ce volumes. Many kinds of equipment and scanners were used to collect samples from many different test centers to have diverse dataset. The dataset was pre-segmented manually by the contributors by one to four raters, following the same annotation protocol, and their annotations were approved by experienced neuro-radiologists [30]. The total number of volumes in this dataset is 285. Among these 285, 210 volumes are High Grade Glioma (HGG) and 75 volumes are Low Grade Glioma (LGG). Each of the volumes has 155 slices. Each of the volumes of data has a shape of (240, 240, 4) and each of the ground truth has a shape of (240, 240, 1).

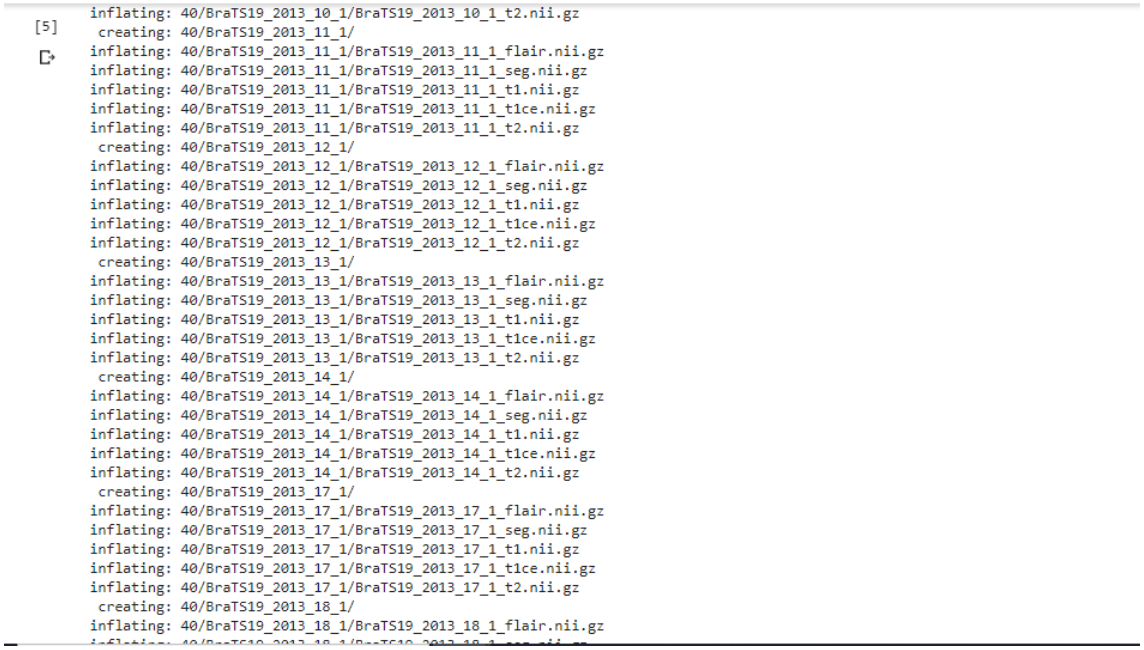

Figure 3.1: Dataset

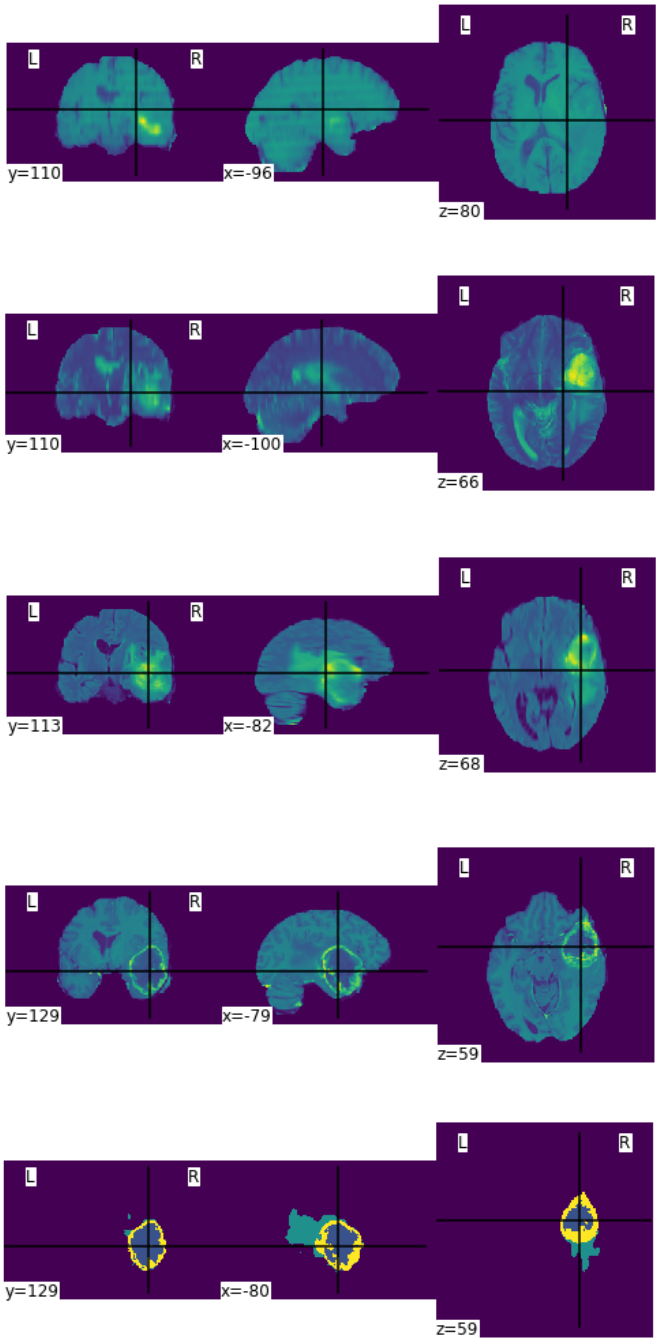

Figure 3.2: From Top To Bottom : T1, T2, Flair, T1Ce, Ground Truth

3.2 Dataset Handling

3.2.1 Reshaping

For U-Net we did reshaping instead of resizing because while resizing it changes the data whereas reshaping keeps the data intact, for better accuracy following the cutting edge approach for biomedical image segmentation.

Initially, the dimension of data was (240, 240, 4) and the dimension of ground truth was (240, 240, 1). Later we reshaped the data to (192, 192, 4) and ground truth to (192, 192, 1).

3.2.2 Splitting

To split the data we used train_test_split library from sklearn in the following parameters -

```
[ ] from sklearn.model_selection import train_test_split
    X_train, X_test, Y_train, Y_test = train_test_split(data, gt, test_size=0.20, random_state=42)
    X_train, X_val, Y_train, Y_val = train_test_split(X_train, Y_train, test_size=0.25,random_state=42)

[ ] print(Y_train.shape)
    print(Y_val.shape)
    print(X_train.shape)
    print(X_test.shape)
    print(X_val.shape)

    (2160, 192, 192, 1)
    (720, 192, 192, 1)
    (2160, 192, 192, 4)
    (720, 192, 192, 4)
    (720, 192, 192, 4)
```

Figure 3.3: Dataset Splitting

After splitting the data into Training and Validation, we passed Y_train and Y_val to 'to_categorical' function. This function does the job of conversion of a class vector to a binary class matrix. We just passed the class vector Y_train and Y_val to this function and didn't pass the total number of classes and data type. And so, the total number of classes will have a value of none and it would consider as the largest number in Y_train and Y_val. Alongside this, it will consider the data type as default float32.

After that, this function will return a binary matrix representation of the input where the axis of the classes will be placed last.

3.3 Implementation of U-Net

Following the architecture of U-Net [12] we implemented the model based on our dataset. We have added the following adjustments to the model -

- We implemented encoder-decoder layer for the U-Net and also added skip connections.
- We have added an additional dropout layer to avoid over-fitting after the convolution layers.
- We have also included ReLU as an activation function for every layer except for the output layer to introduce non-linearity.
- Input Shape = (192,192,4)
- Conv2D: 64 X 3, Stride: 2 X 2, activation: relu
- Conv2D: 128 X 3, Stride: 2 X 2, activation: relu
- Encoder Dropout(0.2)
- Conv2D: 256 X 3, Stride: 2 X 2, activation: relu
- Conv2D: 512 X 3, Stride: 2 X 2, activation: relu
- Conv2D: 1024 X 3, Stride: 2 X 2, activation: relu
- Encoder Dropout(0.2)
- Conv2DTranspose: 1024 X 3, Stride: 2 X 2, activation: relu
- Conv2DTranspose: 512 X 3, Stride: 2 X 2, activation: relu
- Decoder Dropout(0.2)
- Conv2DTranspose: 256 X 3, Stride: 2 X 2, activation: relu
- Conv2DTranspose: 128 X 3, Stride: 2 X 2, activation: relu
- Conv2D: 64 X 3, Stride: 2 X 2, activation: relu
- Decoder Dropout(0.2)
- output = Conv2D: 4 X 1, activation: softmax

After building the model, we compiled the data using our dice coefficient method. Then we fit the model on the training data keeping the batch size 32. Once the number of given epochs are completed, our model would be trained completely and will be available to predict on our test data. For prediction, we developed our prediction method which takes actual data and ground truth as parameters. As test data is split into slices and prediction is generated on each slice which has a shape of [192,92,4]. We compress the predicted mask to a single channel using the argmax function of NumPy library. As ours is a segmentation task and so to determine the accuracy of the segmented mask, we have used dice coefficient as a metric for determining the loss by passing on the segmented mask and ground truth of each slice to our dice coefficient method as parameters which was initially used for the training purpose. Alongside this, we also converted the number of classes of the ground truth to 4 from 3 to be similar to that of the segmentation mask. Finally, we print the segmentation masks compressed in a single channel along with slices of actual data and test data.

Model summary of U-Net based on our adjustments:

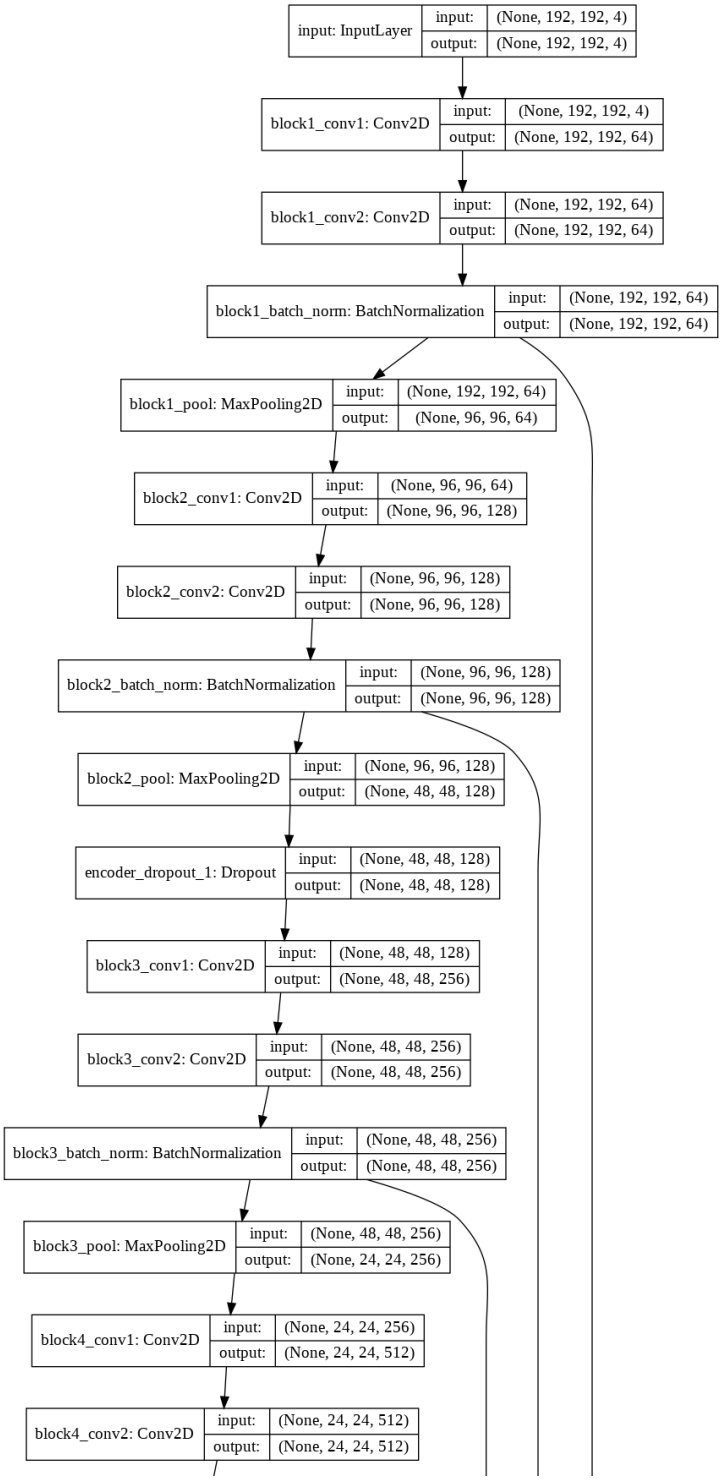

Figure 3.4: U-Net Model Summary Part I

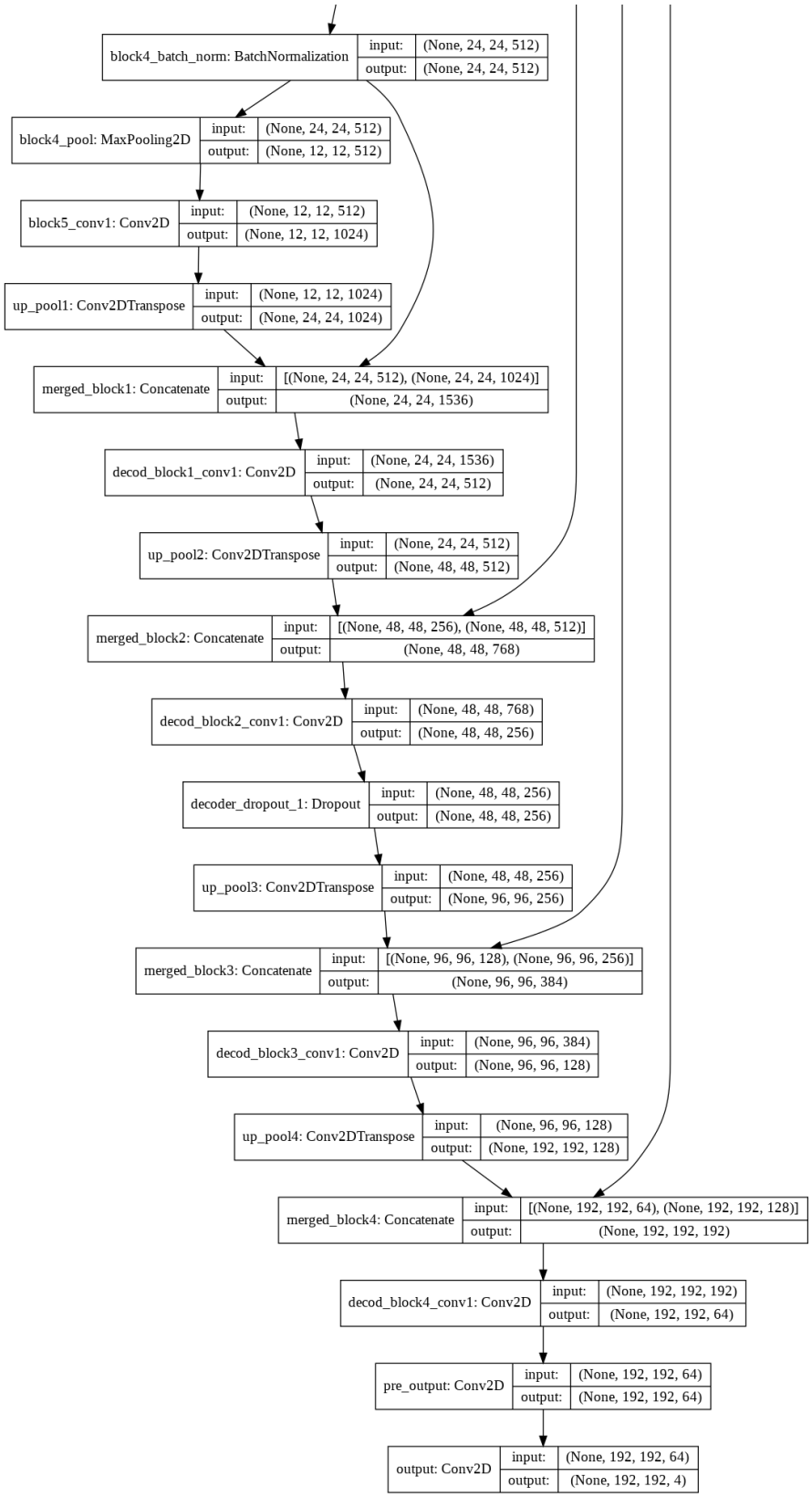

Figure 3.5: U-Net Model Summary Part II

```
for i in range(len(X_test)):
    temp = X_test[i].reshape([-1,192,192,4])
    Y_pre = np.argmax(model.predict((temp)),axis=-1)
    #Y_pre = model.predict((Y_pre))
    np.unique(Y_pre)
    print("Input :", X_test[i].shape)
    print("Output Pre:", Y_pre.shape)
    Y_pre=Y_pre.reshape(-1,192,192,1)
```

Figure 3.6: U-Net Code Snippet Part I

```
    print('X_test '+ str(i))
    plt.imshow(X_test[i,:,:,2])
    plt.show()
    print('Predicted '+ str(i))
    plt.imshow(Y_pre[0,:,:,0])
    plt.show()
    print('Actual '+ str(i))
    plt.imshow(Y_test[i,:,:,0])
    plt.show()
```

Figure 3.7: U-Net Code Snippet Part II

```
    Y_temp = Y_test[i].reshape([-1,192,192,1])
    Y_test_encod = to_categorical(Y_temp,num_classes = 4)

    print("Y Test Encode: " , Y_test_encod.shape)

    score = model.evaluate(temp,Y_test_encod,verbose=0)
    print(model.metrics_names)
    print(score)

    print("Dice Value: " , dice_coef_value(np.array(model.predict((temp)))
                                            , np.array(Y_test_encod)))
```

Figure 3.8: U-Net Code Snippet Part III

3.4 Implementation of Active Learning on U-Net

To implement Active Learning, firstly, we introduced two new methods. One is for calculating Shannon Entropy and the other one is for Batch Processing the dataset. We started with 200 data and provided 60 data at the beginning and left out the other 140 data. After that, we would perform prediction on the remaining datasets and would calculate the Shannon Entropy to find out the highest values. These values signify that the model performed worst on these datasets. Upon receiving the highest entropy values, we identified them and fit the model with these data. This will ensure that our model gets to train again with its worst Shannon Entropy values which result in a better prediction on the next iteration. Following this approach, after each iteration, the model gets to be trained with its worst performed data as well as ensuring that the model sees all the data which differs from traditional approaches where users do not have any control over the data. This approach is often known as 'Black Box'. Undoubtedly, through our proposed approach, we received a result which had better accuracy and prediction.

First of all, we fixed 200 datasets for Active Learning based Training Schedule but unlike the traditional end-to-end approach, we did not fit the whole dataset at once. To initiate the model, we started with 60 datasets and compiled the model as well as performed the first model.fit() function to start the training. To note on this point, the model was being fitted on the model we compiled earlier. Here Adam Optimizer was used and the learning rate was $1 \times e^{-5}$. Moreover, we used the Accuracy metric of the Keras library and for the loss, we manually defined the dice coefficient loss.

```
def dice_loss2(y_true, y_pred, epsilon=1e-6):

    intersection = K.sum(K.abs(y_true * y_pred) , axis = [-2, -3])

    dice = (2. * intersection + epsilon) / (K.sum(K.square(y_true) ,
        axis = [-2, -3]) + K.sum(K.square(y_pred) , axis = [-2, -3]) + epsilon)

    loss = -K.sum(K.log(dice), axis = -1)

    return loss
```

Figure 3.9: Code Snippet of Dice Coefficient Loss

```
model.compile(optimizer=Adam(lr=1e-5),loss=dice_loss2,metrics=['acc'])
```

```
history = model.fit(X_train,Y_train,validation_data=(X_val,Y_val),batch_size=4,epochs=2,shuffle=True)
```

```
WARNING:tensorflow:From C:\Anaconda3\envs\thesis\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.global_
variables is deprecated. Please use tf.compat.v1.global_variables instead.

Train on 2808 samples, validate on 702 samples
Epoch 1/2
2808/2808 [==============================] - 388s 138ms/step - loss: 14.5409 - acc: 0.9200 - val_loss: 21.5913 - val_acc: 0.966
5
Epoch 2/2
2808/2808 [==============================] - 380s 135ms/step - loss: 7.5303 - acc: 0.9747 - val_loss: 17.2845 - val_acc: 0.9670
```

Figure 3.10: Code snippet of Initial Model Fit

Furthermore, after the initial fit, we passed the remaining dataset under batch processing where Shannon Entropy was being calculated. Here we reshaped the dataset again to [-1,192,192,4] and the ground truth to [-1,192,192,1] as we were taking an individual slice from each data (volume).

```
def batch_process(X_train , Y_train):
    list2 = load_list(path2)
    sum_entropy = []
    if(len(global_index)==0):
        for i in range (len(list2)):
            data,gt = load_data_test(path2, list2[i])
            entropy = shanon_method(data, gt)
            sum_entropy.append(entropy)
            print("Entropy: " , entropy)

    else:
        for i in range (len(global_index)):
            for j in (list2):
                if global_index[i] == j:
                    print(global_index[i])
                    print(j)
                    list2.pop(list2.index(j))
```

```
        for i in range (len(list2)):
            print("i " , i)
            print("LIST 2 :" , list2[i])
            data,gt = load_data_test(path2, list2[i])
            entropy = shanon_method(data, gt)
            sum_entropy.append(entropy)
            print("Entropy: " , entropy)

    x = sum_entropy.copy()
    x.sort()
    index_shanon = []
    for i in range (len(sum_entropy)-1 , len(sum_entropy) - 21, -1):
        print(i)
        index_shanon.append(sum_entropy.index(x[i]))

    new_list = []
```

```
    for i in range (len(index_shanon)):
        new_list.append(list2[index_shanon[i]])
        global_index.append(list2[index_shanon[i]])

    print(new_list)

    X_train = []
    Y_train = []


    X_train,Y_train = load_data1(path2, new_list)

    X_train = X_train[:,30:120,30:222,30:222,:].reshape([-1,192,192,4])
    Y_train = Y_train[:,30:120,30:222,30:222].reshape([-1,192,192,1])

    Y_train[np.where(Y_train==4)]=3
```

Figure 3.11: Code snippet of Batch Processing Method Part I

```
    print("Reshape X: " , X_train.shape)
    print("Reshape Y: " , Y_train.shape)


    X_train , Y_train = X_train, Y_train
    X_train, X_val, Y_train, Y_val = train_test_split(X_train, Y_train,
                                            test_size=0.20,shuffle = False)

    Y_train = to_categorical(Y_train)
    Y_val = to_categorical(Y_val)
    X_train = (X_train-np.mean(X_train))/np.max(X_train)
    X_val = (X_val-np.mean(X_val))/np.max(X_val)


    print("Cat X: " , X_train.shape)
    print("cat Y: " , Y_train.shape)

    return X_train , Y_train, X_val, Y_val
```

Figure 3.12: Code snippet of Batch Processing Method Part II

Moreover, before calculating the entropy, the prediction was done over the training data by model.predict() and we called the argmax values over axis = -1. In addition to that, the predicted slice was then encoded by calling 'to_categorical' method of Keras where, number of classes were 4 [to_categorical(Y_temp,num_classes = 4)]. Shanon Entropy was calculated with the following formula: $shanon\_entropy = \sum (predictions \times \log(predictions + \epsilon) + (1 - predictions) \times \log(1 - predictions + \epsilon))$. Here ϵ is $1 \times e^{-12}$.

```
def shanon_method(X_test, Y_test):

    entropy = 0

    X_test = X_test[:,30:120,30:222,30:222,:].reshape([-1,192,192,4])
    Y_test = Y_test[:,30:120,30:222,30:222].reshape([-1,192,192,1])

    print(X_test.shape)
    print(Y_test.shape)

    shanon_list = []

    for i in range(len(X_test)):
        temp = X_test[i].reshape([-1,192,192,4])
        Y_pre = np.argmax(model.predict((temp)),axis=-1)

        np.unique(Y_pre)

        Y_pre=Y_pre.reshape(-1,192,192,1)
        Y_temp = Y_test[i].reshape([-1,192,192,1])
        Y_test_encod = to_categorical(Y_temp,num_classes = 4)

        predictions = model.predict(temp)
        shannon = -np.sum( predictions*np.log2(predictions + 0.000000000001 ) + (1-predictions)
                          *np.log2(1-predictions + 0.000000000001))

        entropy = entropy + (shannon)

    return entropy
```

Figure 3.13: Code snippet of Shanon Entropy Method

Upon receiving the entropy value we stored them in a list and this process was continued for the rest of the data. After calculating the values of Entropy, our next task was to sort the highest values and map them with the original dataset. Moreover, after successful identification of targeted data which have the highest-ranked entropy at that particular iteration, the batch process method returned the targeted data. Later, the model was trained on the received dataset. Therefore, this process was continued seven times to train all of our data.

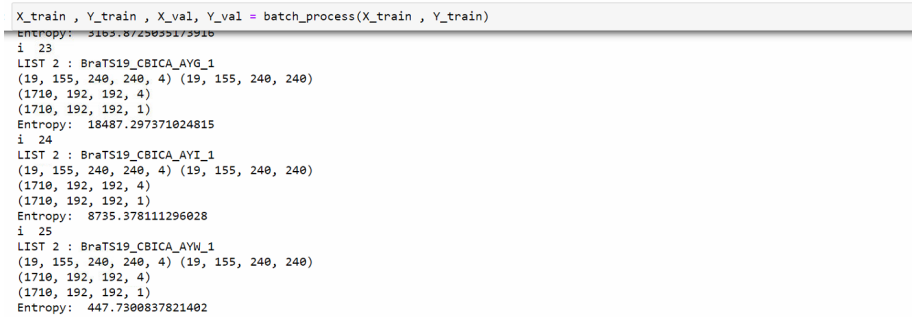

Figure 3.14: Code snippet of Batch Processing and calculating Shannon Entropy

3.5 Activation Function, Optimization Algorithm and Loss Function

While building the U-Net model, we tried to use different activation functions such as Sigmoid, Softmax, ReLU for our model to avoid the outputs which are linear as function and a one-degree polynomial. For our segmentation tasks, ReLU performed the best as it diminishes the problem of gradient vanishing which we noticed while comparing by using other activation functions. Alongside this, ReLU does not require computing any exponential terms which lets us having faster computation. However, in the last layer of our model which is the output layer, Softmax was used as our activation function.

As it is required for us to predict whether our biomedical images contain any modality or not, the probability of positive and negative results should sum up to make the total probability one. Due to this reason, we used Softmax so that we can have a result which sums up to 1. This approach provided us a much better result.

We also have tried many Optimizers such as Stochastic Gradient Descent (SGD), RMSprop but Adam Optimizer which is based on stochastic gradient descent performed the best. As we have used a complex dataset and our model had a too high number of parameters, Adam Optimizer served our purpose most efficiently because of being computationally efficient and consuming less memory.

Dice coefficient is a common metric for pixel segmentation as it can also be modified to act as a loss function. We have used this metric and modified this to use it as a loss function. This is how we calculated the loss function :

$$intersection = \sum predicted \times ground\_truth$$

$$dice = \frac{(2 \times (intersection + \epsilon))}{\sum (predicted^2 + ground\_truth^2)}$$

$$loss = -\sum \log(dice)$$

$$Here,\ ground\_truth\ =\ Ground\ truth\ class,\ predicted\ =\ Predicted\ class$$

Chapter 4

Result Analysis

4.1 U-Net Without Active Learning on BraTS Dataset

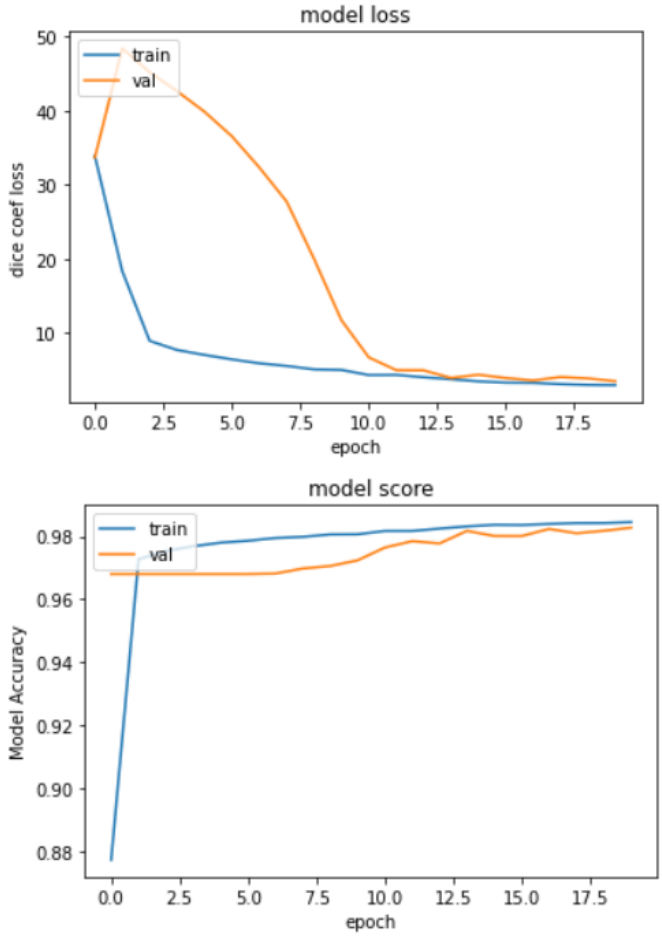

Figure 4.1: U-Net Without Active Learning

U-Net has always been a benchmark model for medical image analysis, especially for 2D data. But in our approach, we have used this Deep Learning model on our complex 3D data by making the necessary pre-processing of the dataset and structuring of the model. If we look at the first graph of Figure 4.1 titled 'model loss'

we can have a better interpretation of how well the model has been trained on our dataset. In the graph, we notice a downfall in the line representing the loss for the training. This denotes that as epoch is passing by, the training loss based on the dice coefficient is reducing. A gradual fall is noticed from the second epoch to the twelfth epoch. During the nineteenth epoch, the loss is almost within the range 1-1.5. The reducing loss value signifies the effective training of the model on our dataset.

Furthermore, if we focus on the second graph of Figure 4.1 titled 'model score' we can see an epoch vs accuracy plot which emphasizes how well the training accuracy has been during the training epochs. As the epoch keeps on increasing the graph also moves towards an upward slope suggesting a higher model accuracy. If we further increase the number of epochs, we might notice our model to overfit which is never desired.

If we then take our segmentation masks into consideration, we can focus on the prediction accuracy of the U-Net model. As ours is a segmentation problem, each mask that is being printed is a superposition of four individual masks which signifies four different regions of tumor prevalent in our dataset. For this purpose, we have a parameter named 'Dice Value' which consists of four values each of which represents the accuracy of the model in determining a particular tumor region for each mask among the four superimposed masks. The different color of the detected modalities in the printed segmentation mask represents the density or structure of a particular tumor region. From the masks predicted we see it's close enough to its actual slice and the value of the prediction loss also provides enough evidence to this cause that the model has predicted what it should to a good extent. Though there are instances where the model could not predict what it should in a slice but all the parameters and metrics such as the loss, accuracy and dice value have been consistent accordingly.

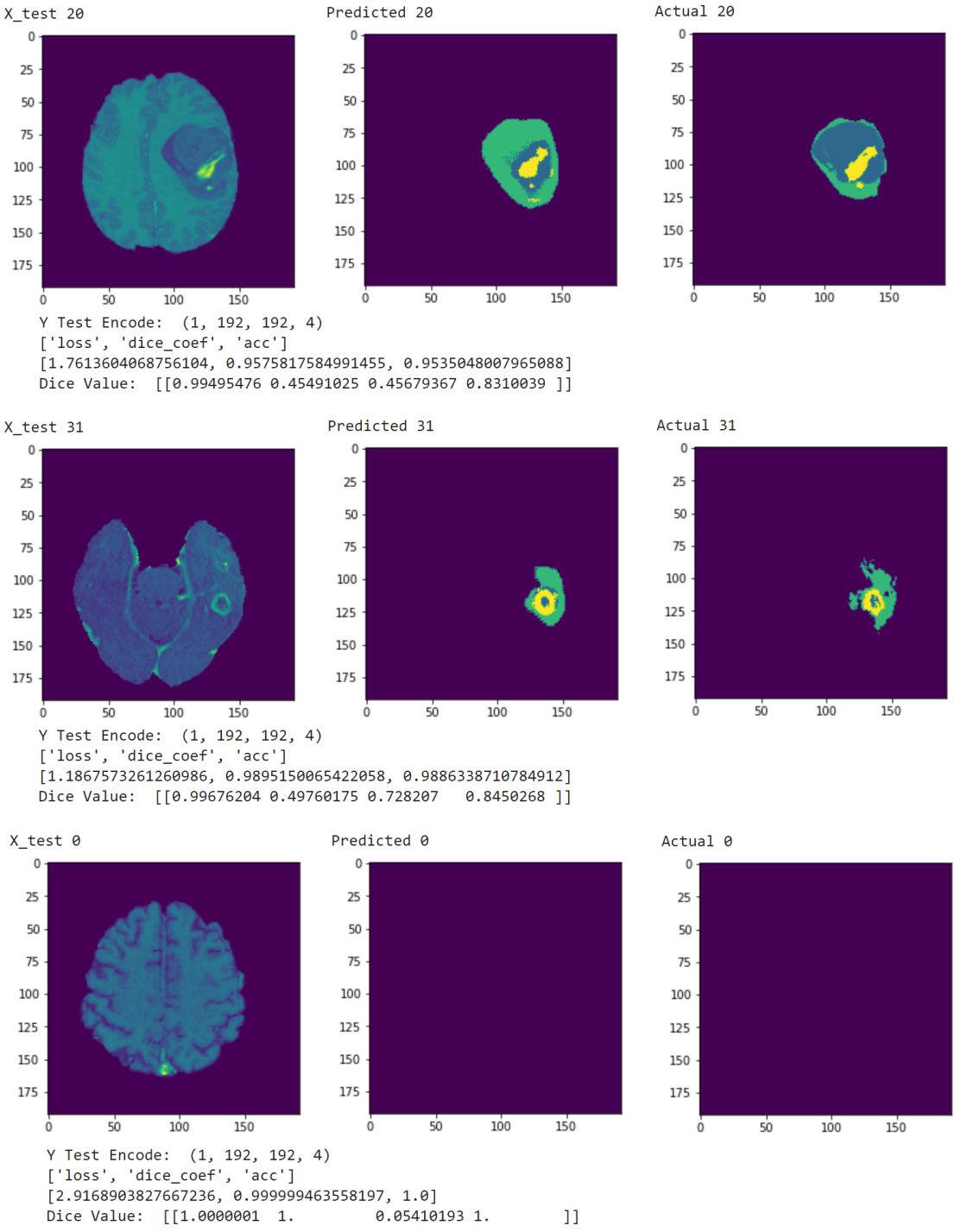

Figure 4.2: U-Net Without Active Learning Slices

4.2 U-Net With Active Learning on BraTS Dataset

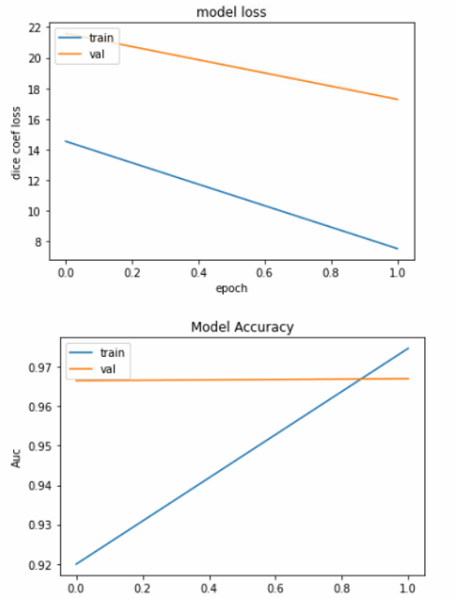

Figure 4.3: Before starting Active Learning

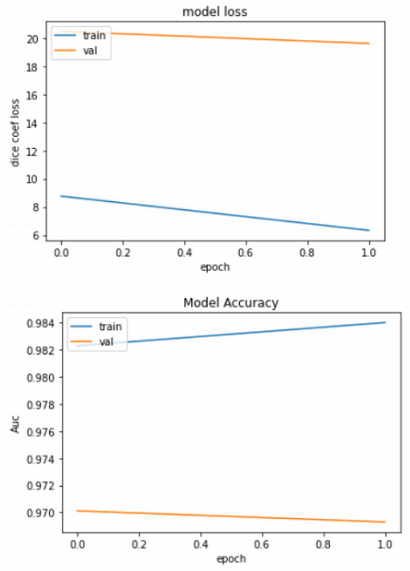

Figure 4.4: Active Learning State 01

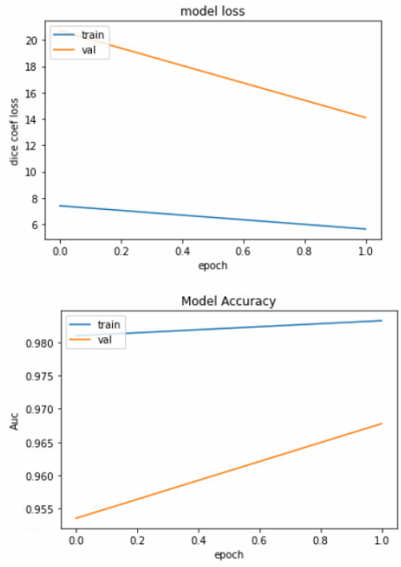

Figure 4.5: Active Learning State 02

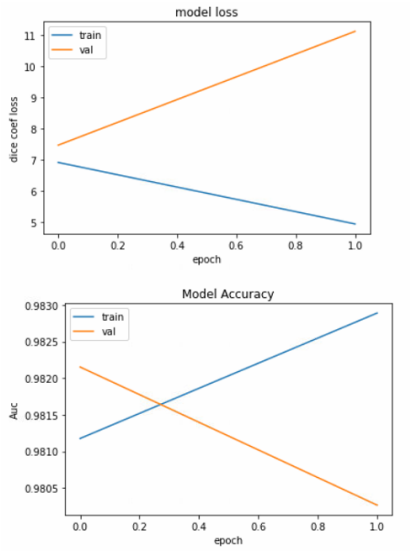

Figure 4.6: Active Learning State 03

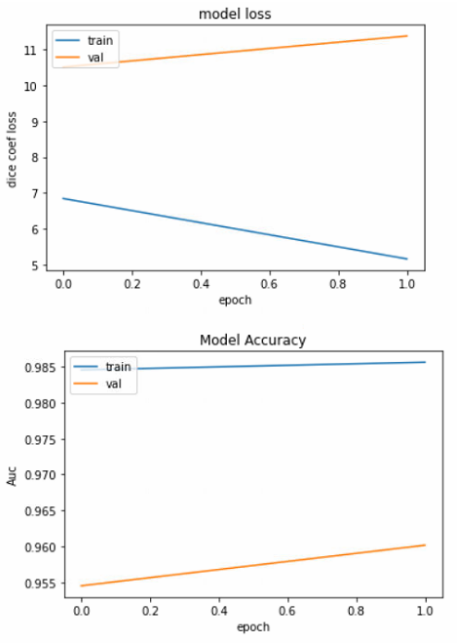

Figure 4.7: Active Learning State 04

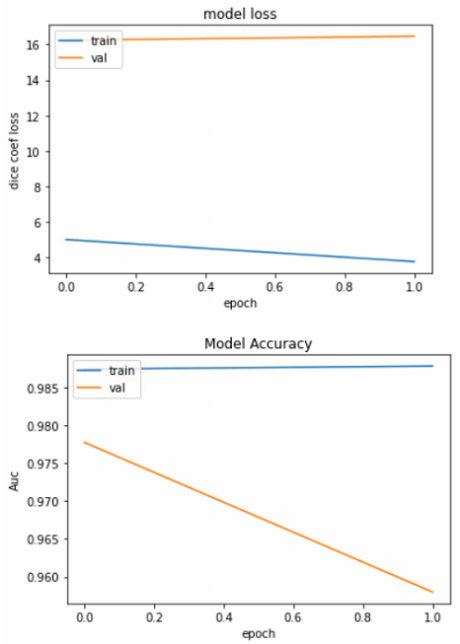

Figure 4.8: Active Learning State 05

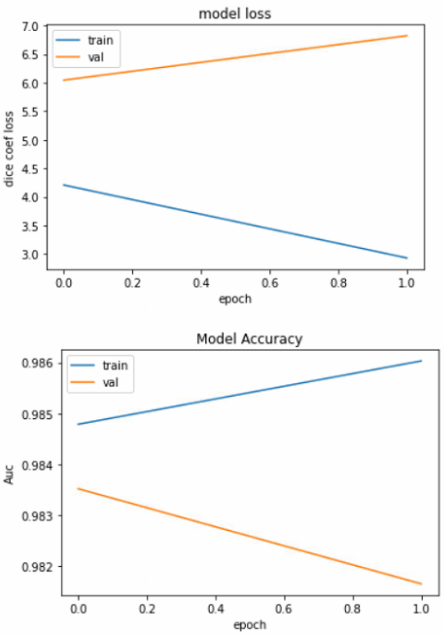

Figure 4.9: Active Learning State 06

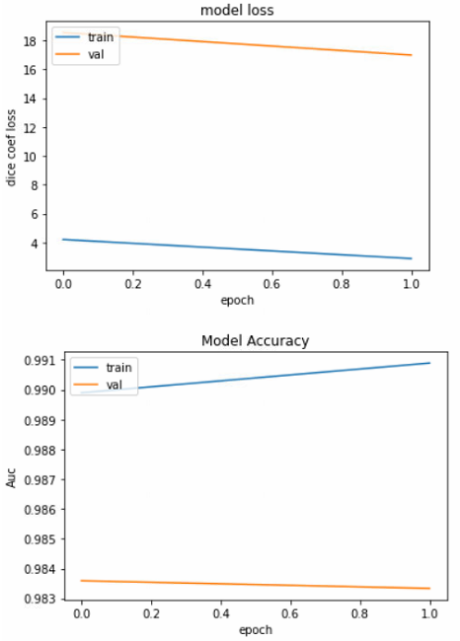

Figure 4.10: Active Learning State 07

For Active Learning based Training Schedule approach, we have trained the dataset with our proposed workflow where the model was trained seven times and for each iteration, there were two epochs. Firstly, to initiate the model we fitted with sixty data and it provided a dice coefficient loss of 7.53 after two epochs (Figure 4.3). This is before implementing Active Learning. After implementing Active Learning based Training Schedule approach, this result drastically improved and Shannon Entropy got reduced too.

From the graphs, it can be seen that after the starting of Active Learning, the loss was 6.3318 (Figure 4.4 : State 01). As we moved further to the next state of Active Learning the model got improved and provided a loss of 5.6243 and an accuracy of 0.9832, validation loss and accuracy were 14.0989 and 96.78% respectively (Figure 4.5 : State 02). Moreover, as the training continued further the model generated a better result. Finally, the loss reduced from 4.9414 (Figure 4.6 : State 03) to 2.8850 (Figure 4.10 : State 07). The accuracy resulted at 99.09% with a validation accuracy of 98.33% after the completion of Active Learning (Figure 4.10 : State 07).

If we look at the predicted segmented values the results are marginally better than the Vanilla U-Net or in other words U-Net without Active Learning. Not only the accuracy got increased but also the Shannon Entropy got decreased drastically. It is also visible that the output of Active Learning developed a very high accurate predicted mask along with higher dice coefficient values. In the segmented predicted masks, we generated four dice values as our dataset is consists of four masks. Each value represents the dice coefficient value of individual masks. As a result, the predicted masks and the ground truth values are intersecting with a high region which is producing a better prediction.

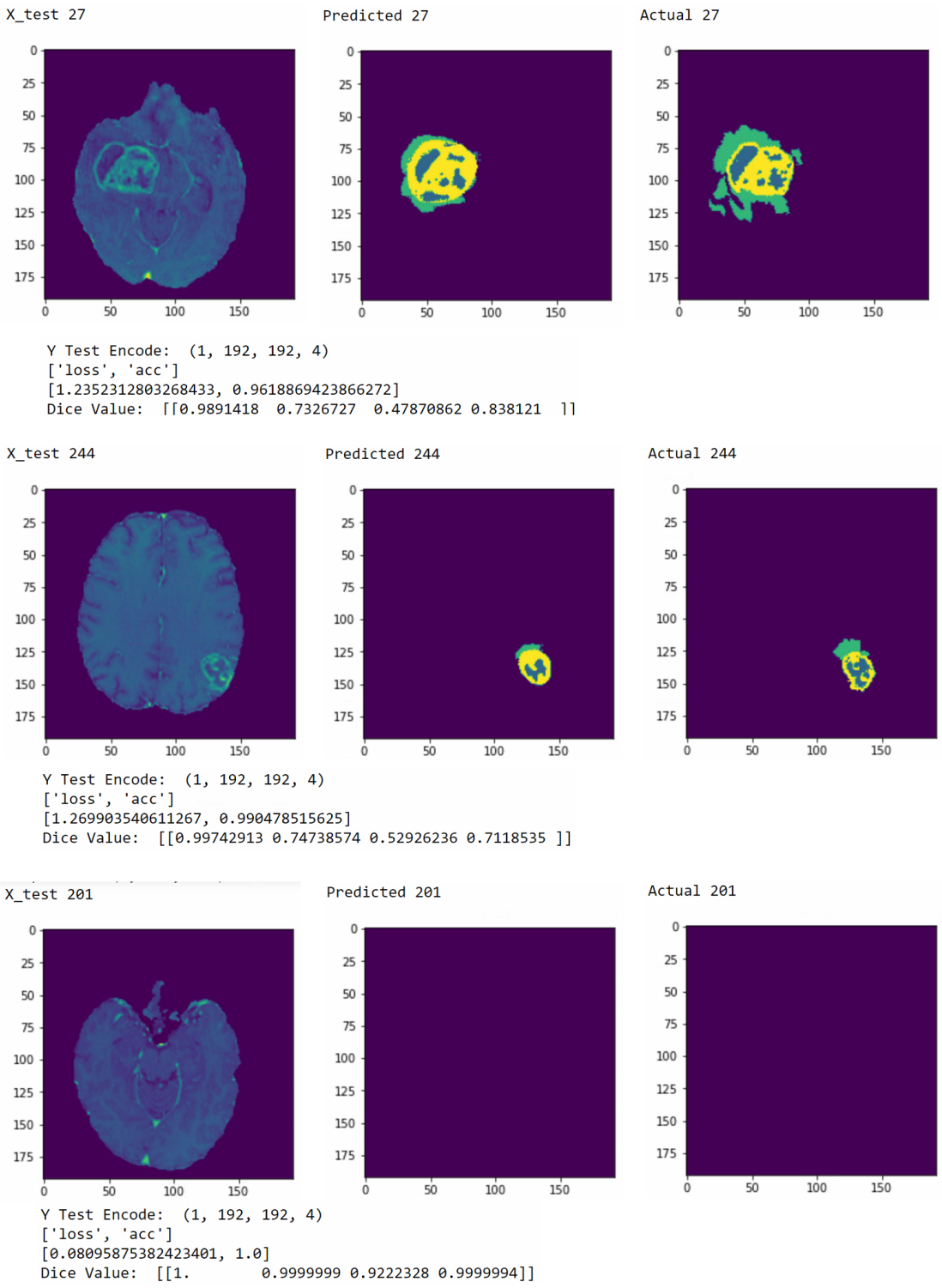

Figure 4.11: U-Net With Active Learning Slices

4.3 Comparison and Result Analysis

4.3.1 Shannon Entropy

Dataset	BraTS19_CBICA_ATF_1	BraTS19_CBICA_AVV_1	BraTS19_TCIA06_247_1	BraTS19_CBICA_AXM_1
Active Learning 01	1153.6475	959.5673	815.5281	906.5152
Active Learning 02	1882.6569	859.7159	1122.1333	1402.2199
Active Learning 03	1602.2217	3053.8707	1071.1515	1240.2884
Active Learning 04	3866.1529	1372.6612	2633.9089	1936.5166
Active Learning 05	1329.2917	859.7159	1621.6286	1736.4798
Active Learning 06	1477.9235	401.3045	1312.9021	782.9043
Active Learning 07	616.80335	301.9062	687.4751	829.1513

Table 4.1: Entropy

Shannon entropy is an uncertainty metric for implementing active learning in our proposed approach. We have calculated the Shannon entropy of all the 200 3D data we had. For the convenience of understanding how this entropy is playing the role of an uncertainty metric, we have highlighted the entropy values of some particular data in table 4.1. The table generalizes the relation between the active learning batches and the corresponding value of some particular data in those batches. For the data BraTS19_CBICA_ATF1 in the initial active learning batch step the value of entropy was 1153.6475 and as the active learning batch step kept on increasing the value of entropy was gradually on the decreasing side. By the end of the seventh active learning batch step the value of entropy was dropped to 616.80335 which is the lowest for that particular data. This signifies that the uncertainty of the data is decreasing.

4.3.2 Active Learning Results after each Iteration

Active Learning State	Loss	Val_Loss	Acc	Val_Acc
01	6.3318	19.6168	98.40	96.93
02	5.6243	14.0989	98.32	96.78
03	4.9414	11.1316	98.29	98.03
04	5.1510	11.3842	98.56	96.01
05	3.7509	16.4653	98.79	95.80
06	2.9289	16.9809	98.60	98.16
07	2.8850	6.8227	99.09	98.33

Table 4.2: Active Learning Results After Each Iteration

While implementing Active Learning the loss after each iteration kept on decreasing which can be seen from Table 4.2. Not only the loss is decreasing but the accuracy is also on the increasing note. The most evident change that we observed after implementing U-Net is that both the training and prediction accuracy have drastically increased. Traditional U-Net implementation took around 17 epochs to attain an acceptable accuracy for the model whereas after implementing our Active Learning based Training Schedule model, it is attaining similar or more accuracy within two epochs.

4.3.3 Comparison with Model Loss and Model Accuracy

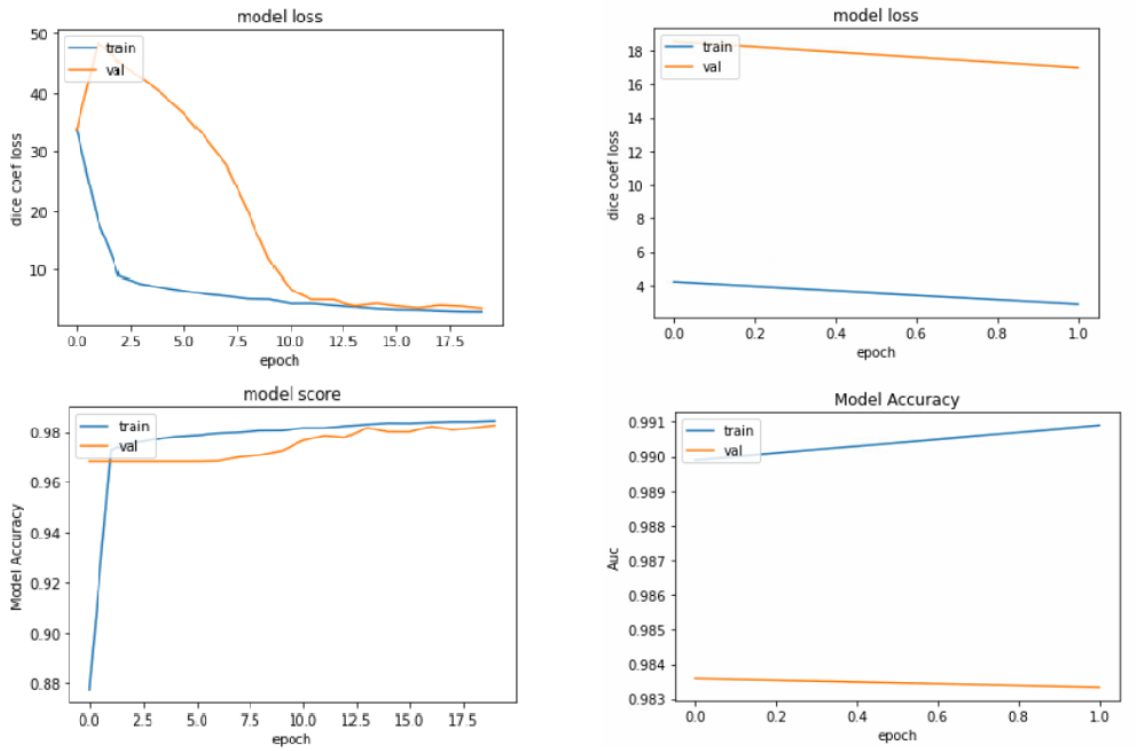

Figure 4.12: Comparison

It is not only limited to the training accuracy as we can see the segmentation masks that have been predicted after Active Learning being implemented is much better predicted than that of the traditional vanilla U-Net prediction. This can be verified from Figure 4.11 where we can see much better predicted mask have been plotted than that of U-Net in terms of dice coefficient loss and accuracy. The prediction accuracy value of each mask of all four masks being predicted shows the enhancement in different tumor region prediction. Besides, the model has been consistent in terms of loss and accuracy with respect to the predicted and actual slices. Thus, our purpose of introducing a new Active Learning based Training Schedule for Biomedical Image segmentation is being served.

Chapter 5

Conclusion

5.1 Conclusion

We have tried to improve the traditional Deep Learning (DL) based techniques by introducing Active Learning(AL) which can be implemented in several phases of the Deep Learning pipeline during annotation, during training, or during inference. We have implemented an Active Learning based Training Schedule approach and ended up achieving a much better performance than the traditional end-to-end approaches on U-Net for segmenting Brain Tumor of MICCAI BraTS 2019 dataset. Apart from achieving a higher prediction accuracy, we can see the segmentation masks that have been predicted after implementing Active Learning is much better predicted than that of the traditional vanilla U-Net prediction. Besides, the model has been consistent in terms of loss and accuracy with respect to the predicted and actual slices. We are hoping to come up with new methodologies that combine the strengths of Active Learning and Human-in-the-loop (HITL) for the development of Deep Learning based applications that can be used in clinical practice.

5.2 Challenges

As we have worked with large 3D MRI data of Brain Tumors which were around 7GB of biomedical data and so high configuration computing devices (RAM, CPU, GPU etc.) were required. Due to hardware limitations, we couldn't train the whole dataset at once. Apart from that, for a long time, we faced an error called 'ResourceExhaustedError'. This error was also raised due to hardware limitations. We couldn't solve this issue even with the help of Google Colab too as Colab itself has its limitations. After that, we took an approach of training the data by taking an 'on the go' training approach which eventually solved our issue and so we could finish our research work in due time.

5.3 Future Work Plan

In our research, we have only used one model (U-Net). In the future, we intend to try out other different models DoubleU-Net, UNet++ which are other variants of U-Net to incorporate our approach. We are hoping to have significant improvement in performance from these models if we implement Active Learning on them.

Apart from this, we want to improvise the concept of Human-in-the-loop (HITL) to a great extent by including more supervised human intervention in the Active Learning framework.

Bibliography

[1] P. Singh, S. Singh, and G. Kaur, "A study of gaps in cbmir using different methods and prospective," in *Proceedings of world academy of science, engineering and technology*, vol. 36, 2008, pp. 492–496.

[2] D. Erhan, P.-A. Manzagol, Y. Bengio, S. Bengio, and P. Vincent, "The difficulty of training deep architectures and the effect of unsupervised pre-training," in *Artificial Intelligence and Statistics*, 2009, pp. 153–160.

[3] Z. Ma, J. M. R. Tavares, and R. N. Jorge, "A review on the current segmentation algorithms for medical images," in *Proceedings of the 1st International Conference on Imaging Theory and Applications (IMAGAPP)*, 2009.

[4] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.

[5] B. Menze, A. Jakab, S. Bauer, J. Kalpathy-cramer, K. Farahani, J. Kirby, *et al.*, "The multimodal brain tumor image segmentation benchmark (brats). medical imaging," *IEEE Transactions on*, pp. 1–32, 2014.

[6] H. R. Roth, L. Lu, A. Seff, K. M. Cherry, J. Hoffman, S. Wang, J. Liu, E. Turkbey, and R. M. Summers, "A new 2.5 d representation for lymph node detection using random sets of deep convolutional neural network observations," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2014, pp. 520–527.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[9] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, Springer, 2014, pp. 818–833.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[11] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1520–1528.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[13] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2016, pp. 424–432.

[14] R. Golan, C. Jacob, and J. Denzinger, "Lung nodule detection in ct images using deep convolutional neural networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2016, pp. 243–250.

[15] J. Kawahara, A. BenTaieb, and G. Hamarneh, "Deep features to classify skin lesions," in *2016 IEEE 13th international symposium on biomedical imaging (ISBI)*, IEEE, 2016, pp. 1397–1400.

[16] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 Fourth International Conference on 3D Vision (3DV)*, IEEE, 2016, pp. 565–571.

[17] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[18] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-effective active learning for deep image classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 12, pp. 2591–2600, 2016.

[19] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[20] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, J. Freymann, K. Farahani, and C. Davatzikos, "Segmentation labels and radiomic features for the pre-operative scans of the tcga-gbm collection. the cancer imaging archive," *Nat Sci Data*, vol. 4, p. 170 117, 2017.

[21] C. F. Baumgartner, L. M. Koch, M. Pollefeys, and E. Konukoglu, "An exploration of 2d and 3d deep learning techniques for cardiac mr image segmentation," in *International Workshop on Statistical Atlases and Computational Models of the Heart*, Springer, 2017, pp. 111–119.

[22] Q. Dou, L. Yu, H. Chen, Y. Jin, X. Yang, J. Qin, and P.-A. Heng, "3d deeply supervised network for automated segmentation of volumetric medical images," *Medical image analysis*, vol. 41, pp. 40–54, 2017.

[23] X. Feng, J. Yang, A. F. Laine, and E. D. Angelini, "Discriminative localization in cnns for weakly-supervised segmentation of pulmonary nodules," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2017, pp. 568–576.

[24] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," *arXiv preprint arXiv:1703.02910*, 2017.

[25] K. Kamnitsas, C. Baumgartner, C. Ledig, V. Newcombe, J. Simpson, A. Kane, D. Menon, A. Nori, A. Criminisi, D. Rueckert, *et al.*, "Unsupervised domain adaptation in brain lesion segmentation with adversarial networks," in *International conference on information processing in medical imaging*, Springer, 2017, pp. 597–609.

[26] K. Kamnitsas, C. Ledig, V. F. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, and B. Glocker, "Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation," *Medical image analysis*, vol. 36, pp. 61–78, 2017.

[27] D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," *Annual review of biomedical engineering*, vol. 19, pp. 221–248, 2017.

[28] L. Yang, Y. Zhang, J. Chen, S. Zhang, and D. Z. Chen, "Suggestive annotation: A deep active learning framework for biomedical image segmentation," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2017, pp. 399–407.

[29] G. Zeng, X. Yang, J. Li, L. Yu, P.-A. Heng, and G. Zheng, "3d u-net with multi-level deep supervision: Fully automatic segmentation of proximal femur in 3d mr images," in *International workshop on machine learning in medical imaging*, Springer, 2017, pp. 274–282.

[30] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, R. T. Shinohara, C. Berger, S. M. Ha, M. Rozycki, *et al.*, "Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge," *arXiv preprint arXiv:1811.02629*, 2018.

[31] Y. Gordienko, P. Gang, J. Hui, W. Zeng, Y. Kochura, O. Alienin, O. Rokovyi, and S. Stirenko, "Deep learning with lung segmentation and bone shadow exclusion techniques for chest x-ray analysis of lung cancer," in *International Conference on Computer Science, Engineering and Education Applications*, Springer, 2018, pp. 638–647.

[32] M. A. Islam, M. Rochan, S. Naha, N. D. B. Bruce, and Y. Wang, *Gated feedback refinement network for coarse-to-fine dense semantic image labeling*, 2018. arXiv: 1806.11266 [cs.CV].

[33] S. Renukalatha and K. Suresh, "A review on biomedical image analysis," *Biomedical Engineering: Applications, Basis and Communications*, vol. 30, no. 04, p. 1 830 001, 2018.

[34] G. Zeng and G. Zheng, "Multi-stream 3d fcn with multi-scale deep supervision for multi-modality isointense infant brain mr image segmentation," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 2018, pp. 136–140.

[35] Z. Zhang, X. Zhang, C. Peng, X. Xue, and J. Sun, "Exfuse: Enhancing feature fusion for semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 269–284.

[36] S. Budd, E. C. Robinson, and B. Kainz, "A survey on active learning and human-in-the-loop deep learning for medical image analysis," *arXiv preprint arXiv:1910.02923*, 2019.

[37] M. H. Hesamian, W. Jia, X. He, and P. Kennedy, "Deep learning techniques for medical image segmentation: Achievements and challenges," *Journal of digital imaging*, vol. 32, no. 4, pp. 582–596, 2019.

[38] K. Konyushkova, R. Sznitman, and P. Fua, "Geometry in active learning for binary and multi-class image segmentation," *Computer vision and image understanding*, vol. 182, pp. 1–16, 2019.

[39] S. A. Taghanaki, K. Abhishek, J. P. Cohen, J. Cohen-Adad, and G. Hamarneh, "Deep semantic segmentation of natural and medical images: A review," *arXiv preprint arXiv:1910.07655*, 2019.