DL Project 2 Report

Neeraj(B20BB015) Mandheer(B20BB021) Kshitij(B20BB017)

Problem statement

Our aim is to find a tumor using CT scan data or PET scan data, along with dark or light markings that show where the tumor is and how big it is. Based on this information, the following could be said to be true:

- A cancer's grade is based on how big it is. As a result, if a patient follows their doctor's advice and receives regular screenings, it may be possible to determine the grade of the cancer based on its size.
- The quality of the scan and the information it gives us could also help us figure out the stage of the cancer to some extent.

Introduction

A CT scan uses X-rays to create pictures of the inside of your body. The machine takes pictures from different angles, and a computer puts them together to create a detailed image. This can help doctors see if there are any abnormal masses in your body, like tumors. CT scans can also show if the cancer has spread to other parts of your body. An MRI scan, on the other hand, uses magnets and radio waves to create pictures of your body. Doctors can use these pictures to see the soft tissues inside your body and look for any abnormalities. MRI scans can be used to find brain tumors, which may or may not be cancerous.

Scientists have developed computer programs that use CT and MRI images to help doctors diagnose and treat cancer. These programs are called deep learning models, and they use artificial intelligence to analyze the images and make predictions. Some studies have shown that these models can be even more accurate than human doctors in detecting cancer. However, doctors still need to do a biopsy to confirm the diagnosis and determine the best course of treatment. So, while these computer programs are helpful tools, they cannot replace the experience and expertise of a trained medical professional.

Methodology

Data Preprocessing

- We took the dataset from the PICAI website, where there were two datasets present with different file types: the nii.gz file type (labeled image) and the.mha file type (original image). Here, both of the datasets had images with different resolutions.
- We created a class through which we were able to perform a few tasks, specifically downloading the file and saving it at a specified location.
- Then we used the Medpy library to convert the dataset into a numpy array.
- To solve the different resolution of each image problem, we decided to resize the model to have the same dimension for all, which is (50, 104, 104) for each patient, which brings our total numpy array data stored to be (1000, 50, 104, 104).
- In this, we have two sets of images, one with only images and the other with labeled images.
- Finally, we used the np.save function to save the file, which was needed as Medpy is very slow to work with.

Convolution and autoencoder

- To see the localization of the tumor present in the image, we are using an autoencoder.
- At the end of the autoencoder, we attached a linear layer through which we are classifying the ISUP case.
- For the encoder part, we used 3D convolution, in which there are 3 blocks with 3 conv3D layers each.
- Now for the decoder part, we did the same, but here the conv layers in each block had 1 conv3d transpose layer and 2 conv3d layers.

Here we are calculating three different losses:

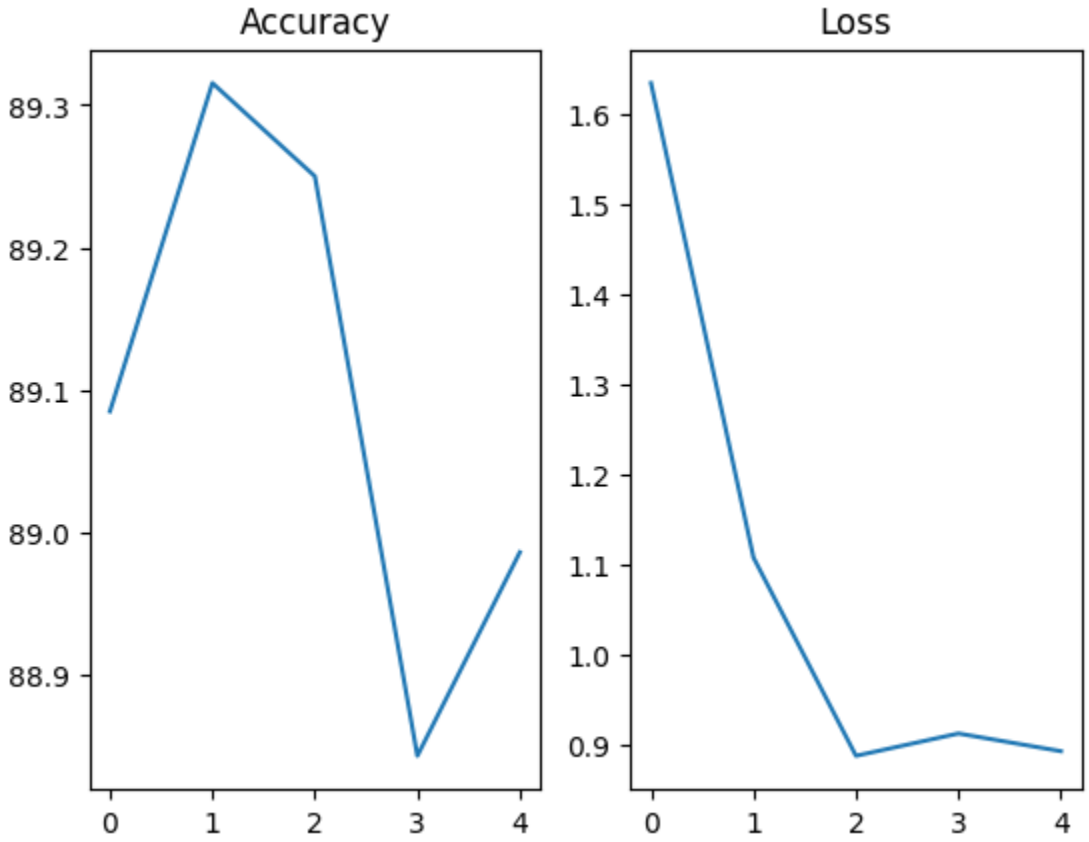
- Reconstruction loss (using the MSE loss function, and we are using this reconstruction loss for the unbiasedness report from the image analysis)
- Tumor labeling loss (is done using the MSE loss function and is done using grayscale, which shows the rest of the image in black but the tumor in white).
- classification loss (uses the cross-entropy function)

UNET model and DICE loss function

- But the MSE loss function is unable to be unbiased as, at the image where there is no tumor present, it is printing "0" as it is supposed to. But due to the fact that the training was done on a dataset that had a smaller tumor size, and as the loss was increasing, it was implying that the image was full black and had no white spots (representing tumors), so in the place where it was supposed to print 1, it is printing it as "0," which is implying that there is no tumor present in the images, due to which we had to again start from the beginning.
- Through great determination of going through each code from different applications to get a reference to start again, the code that was found in github was promising to have

done similar things on the application of brain cancer, but we were able to use its application to our convenience for prostate cancer, which are to specify: - UNET model and DICE Loss function [DICE Loss = 1 dice coefficient; dice coefficient is defined by the ratio of twice the overlapping area of labeled image and raw image to the total number of pixels (labeled image + raw image)].

- We did a few modifications in the UNET to do the classification ISUP case to specify that we attached 1 Maxpool layer followed by 3 linear layers and in between a relu function. We used Downconv4's output to feed it to the liner layer, and from the linear layer's output, we are doing the classification of tumors.
- So, now as our model is ready we can actually start training using our preprocessed data from earlier.
- Training will now begin, from observation if mha file's annotation is completely filled with 0, we basically randomly have a chance of not training it, as only 27% of the dataset has annotations with non zero values.



Reference

- https://github.com/s0mnaths/Brain-Tumor-Segmentation
- https://pi-cai.grand-challenge.org/AIPR/
- https://www.cancerimagingarchive.net/