Information Technology Institute (ITI)

AI/ML Department

Multi-Cancer Identification and Segmentation

Computer Vision Capstone Project

<u>Made by:</u>

1 – Muhammad Khaled Hamza

2 – Yousef Khaled El-Gyoushy

3 – Sherif El-Sayed Sherif

4 – Ashraf Ehab Mohammed

5 – Mohamed Ali Kamal

Table of Contents

1 – Introduction

1 – 1 System Architecture

The goal of our project is to be able to classify and segment tumors from both brain and breast radiology (scans), to be able to execute this we propose the following system architecture:

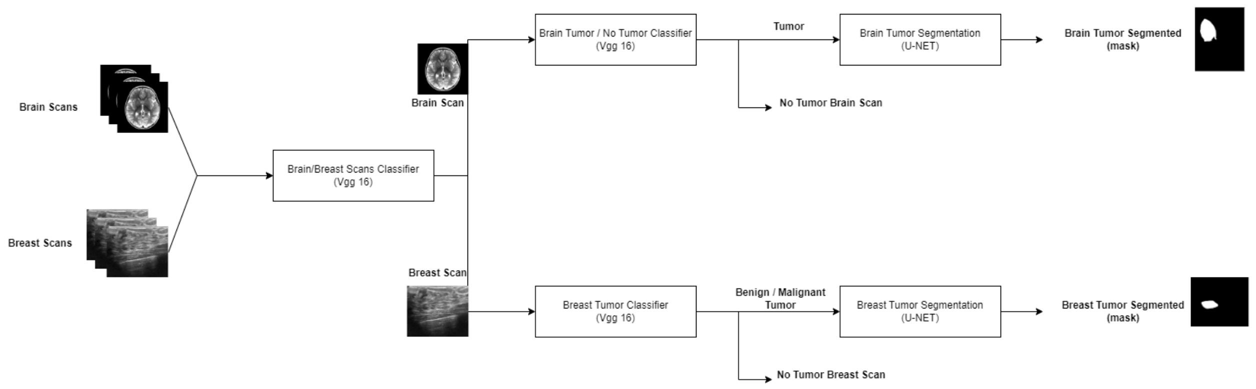

Brain Scans
Breast Scans
Brain/Breast Scans Classifier (Vgg 16)
Brain Scan
Breast Scan
Brain Tumor / No Tumor Classifier (Vgg 16)
Tumor
Brain Tumor Segmentation (U-NET)
Brain Tumor Segmented (mask)
No Tumor Brain Scan
Breast Tumor Classifier (Vgg 16)
Benign / Malignant Tumor
Breast Tumor Segmentation (U-NET)
Breast Tumor Segmented (mask)
No Tumor Breast Scan

Fig 1.1 Proposed System Architecture

1 – 2 Image Processing

As for the different image processing used, we used data augmentation with various image transformations, to reduce overfitting of the different models used and to introduce some variety in our models. The following includes some of the transformations applied on samples of our dataset:

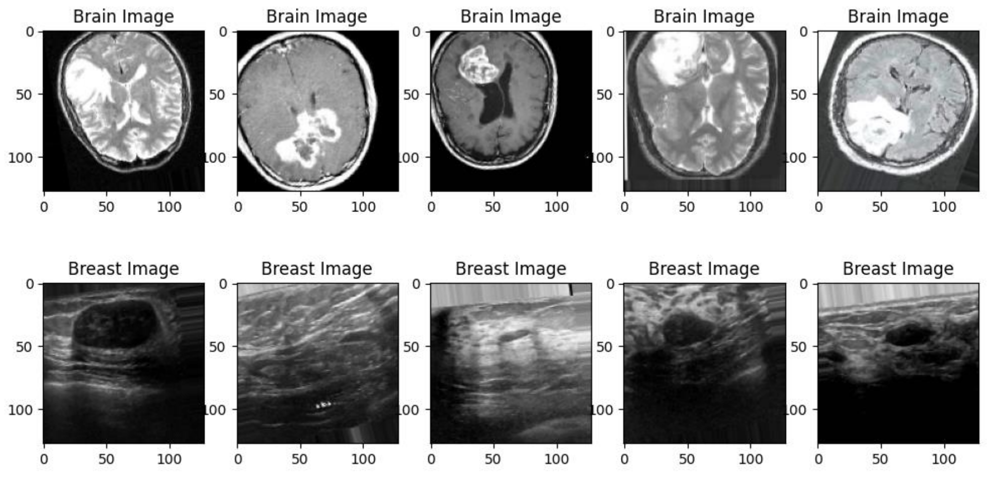
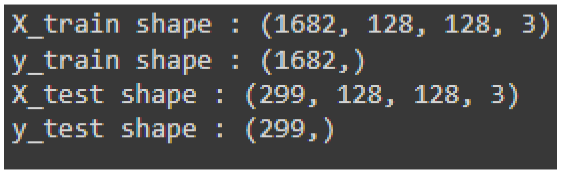


Fig 1.2 Image Processing Applied to The Brain and Breast Scans

2 - Brain/Breast 1st Stage Classification

2 – 1 Data Preparation

In our system the first classifier is brain/breast scans classifier, for this model we have brain and breast scans, after putting the data into the respective train and test arrays various data augmentation techniques are applied to reduce overfitting of the model and to introduce variations.



```
X_train shape : (1682, 128, 128, 3)
y_train shape : (1682,)
X_test shape : (299, 128, 128, 3)
y_test shape : (299,)
```

Fig 2.1 Train and Test Data Shapes

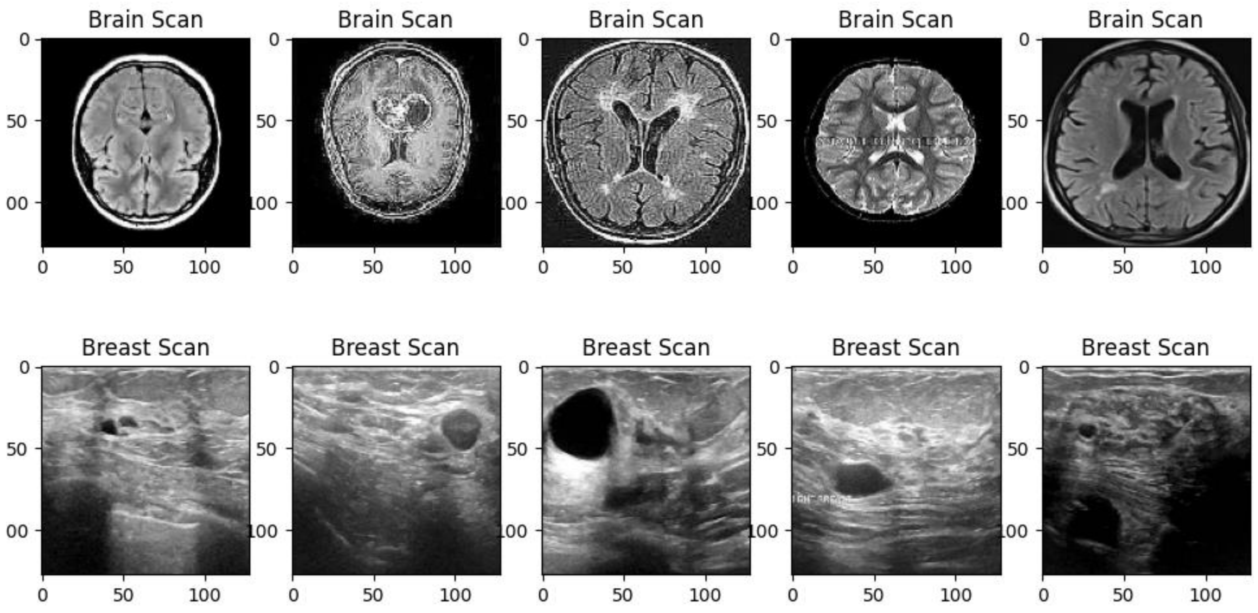


Fig 2.2 Brain and Breast Scans

2 – 2 Proposed Architecture

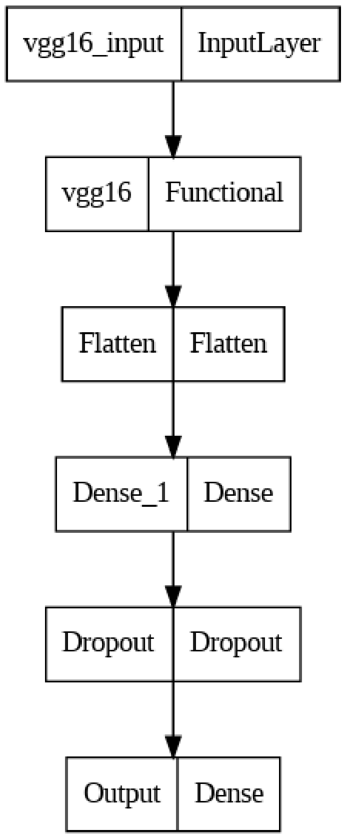


Fig 2.3 VGG16-Based Brain and Breast Classifier

We set up a convolutional neural network (CNN) model for binary image classification, based on the VGG16 architecture with transfer learning.

We used VGG16 model as a base model, where the weights argument is set to 'imagenet' for using pre-trained weights on the ImageNet dataset, include_top argument is set to False for excluding the final classification layer, and input shape argument is set to (128, 128, 3) to specify the size of the input images.

We wanted only want to fine-tune the weights of the fully connected layers that we add on top of the pre-trained model, and not the pre-trained weights themselves, so we froze the VGG16 layers, preventing them from updating their weights

The next layer is a fully connected Dense layer with 256 neurons and a ReLU activation function. This layer learns to extract high-level features from the flattened input.

The next layer is a Dropout layer, which randomly drops out 20% of the neurons in the previous layer during training. This helps prevent overfitting by forcing the network to learn more robust features.

The final layer is another fully connected Dense layer with a single neuron and a sigmoid activation function, which outputs a probability score indicating the likelihood that the input image belongs to the positive class (in this case, a binary classification task).

2 – 3 Training Results

The model was compiled using “binary crossentropy” as our loss function, with Adam optimizer, the model was trained for 5 epochs and took roughly 1 minute to train:

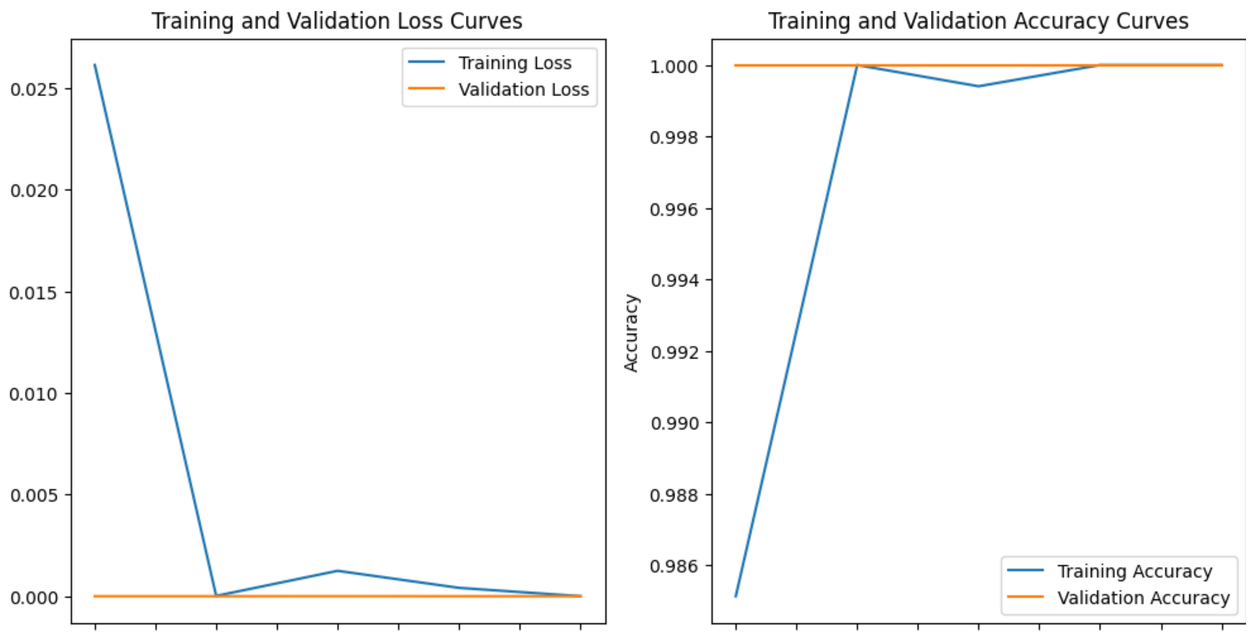
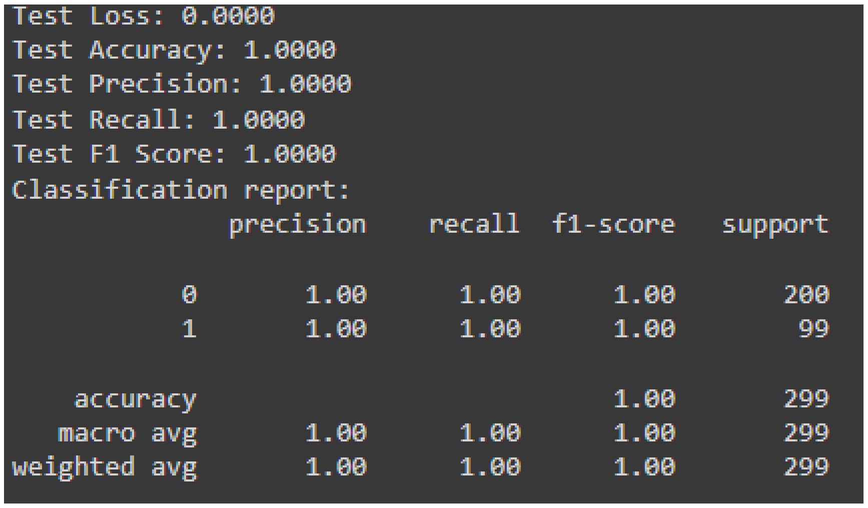


Fig 2.4 VGG16-Based Brain and Breast Classifier Training Results

2 – 4 Testing Metrics



```
Test Loss: 0.0000
Test Accuracy: 1.0000
Test Precision: 1.0000
Test Recall: 1.0000
Test F1 Score: 1.0000
Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       200
           1       1.00      1.00      1.00        99

    accuracy                           1.00       299
   macro avg       1.00      1.00      1.00       299
weighted avg       1.00      1.00      1.00       299
```

Fig 2.5 VGG16-Based Brain and Breast Classifier Testing Results

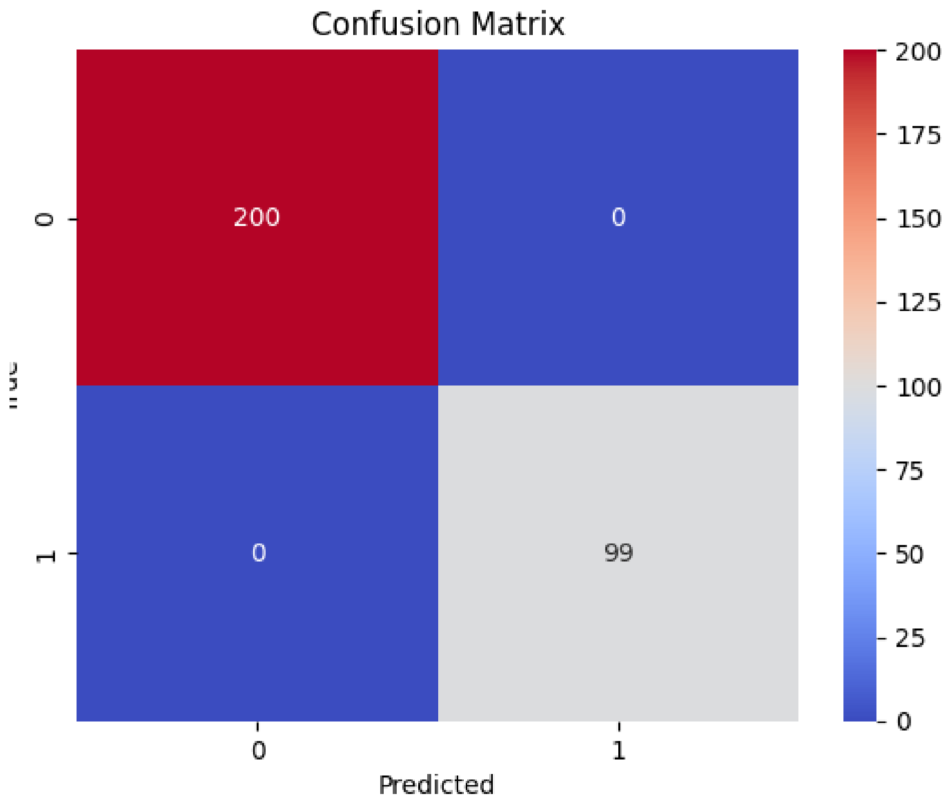


Fig 2.6 VGG16-Based Brain and Breast Classifier Confusion Matrix

We can see that our model did an excellent job in distinguishing brain and breast classifiers apart, since it is a fairly easy task.

2 – 5 Testing Results

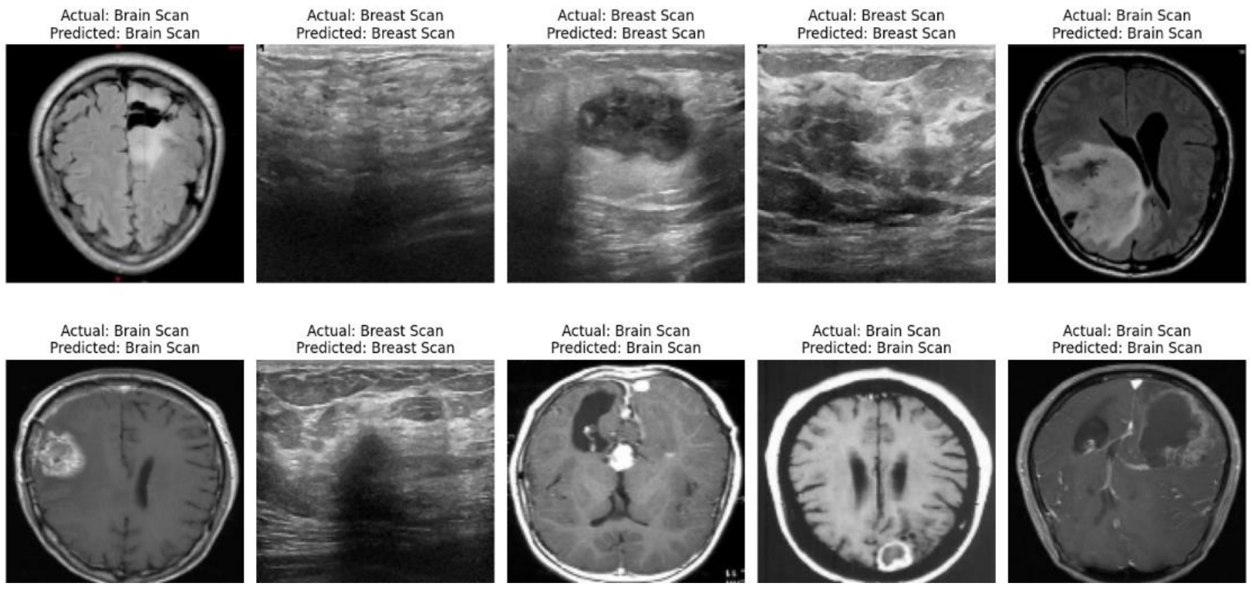


Fig 2.7 VGG16-Based Brain and Breast Classifier Confusion Matrix

As we can see our model succeeded in the task of telling the brain and breast scans apart.

3 – Brain Tumor 2nd Stage Classification

3 – 1 Data Preparation

The second stage is to be able to tell if the scan itself has a tumor our not, for this we will use a classifier to receive a brain scan, and tell us it contains a tumor or not.

Since the dataset is small, some image augmentation techniques were applied to the training dataset to increase the size of the dataset and prevent overfitting. Augmentation techniques are used to generate new training examples by applying random transformations to the existing images. This helps the model generalize better and learn to recognize objects from different angles, scales, and orientations.

The specific augmentations used in this scenario are:

i. Zooming - randomly zooms in or out of the image

ii. Shearing - randomly applies a shearing transformation to the image

iii. Rotation - randomly rotates the image by a certain degree

iv. Horizontal flipping - randomly flips the image horizontally

v. Shifting - randomly shifts the image horizontally or vertically by a certain amount

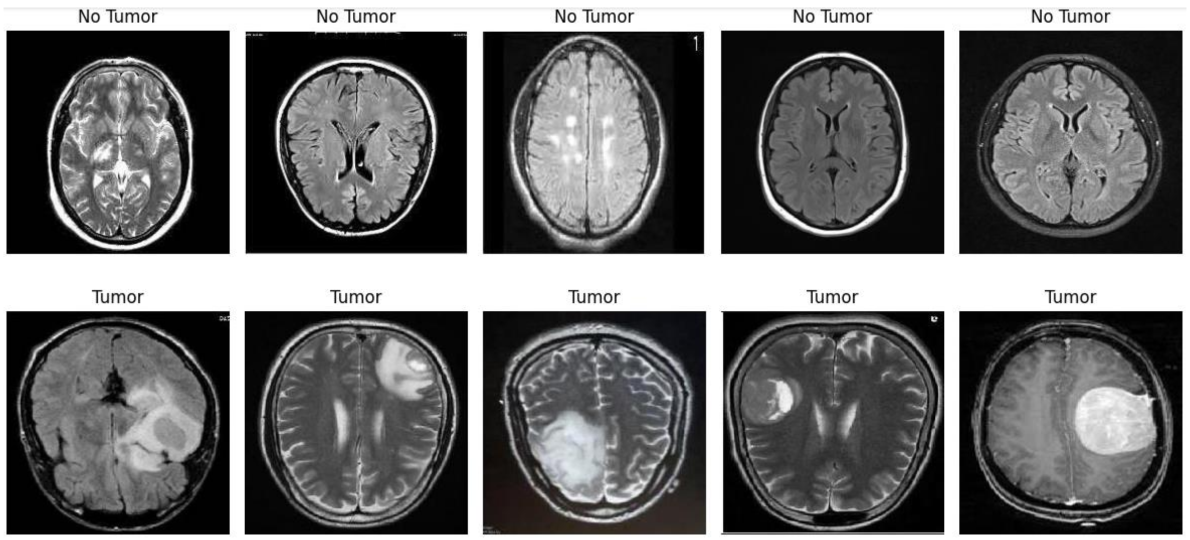


Fig 3.1 Brain Scans Sample

2 – 2 Proposed Architecture

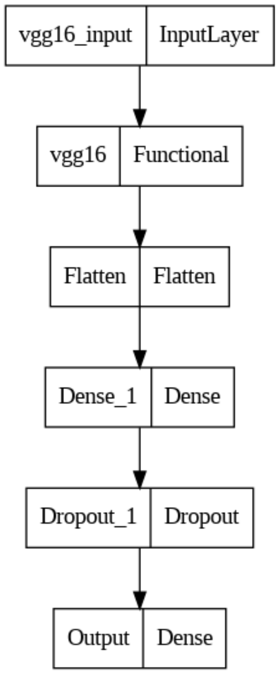
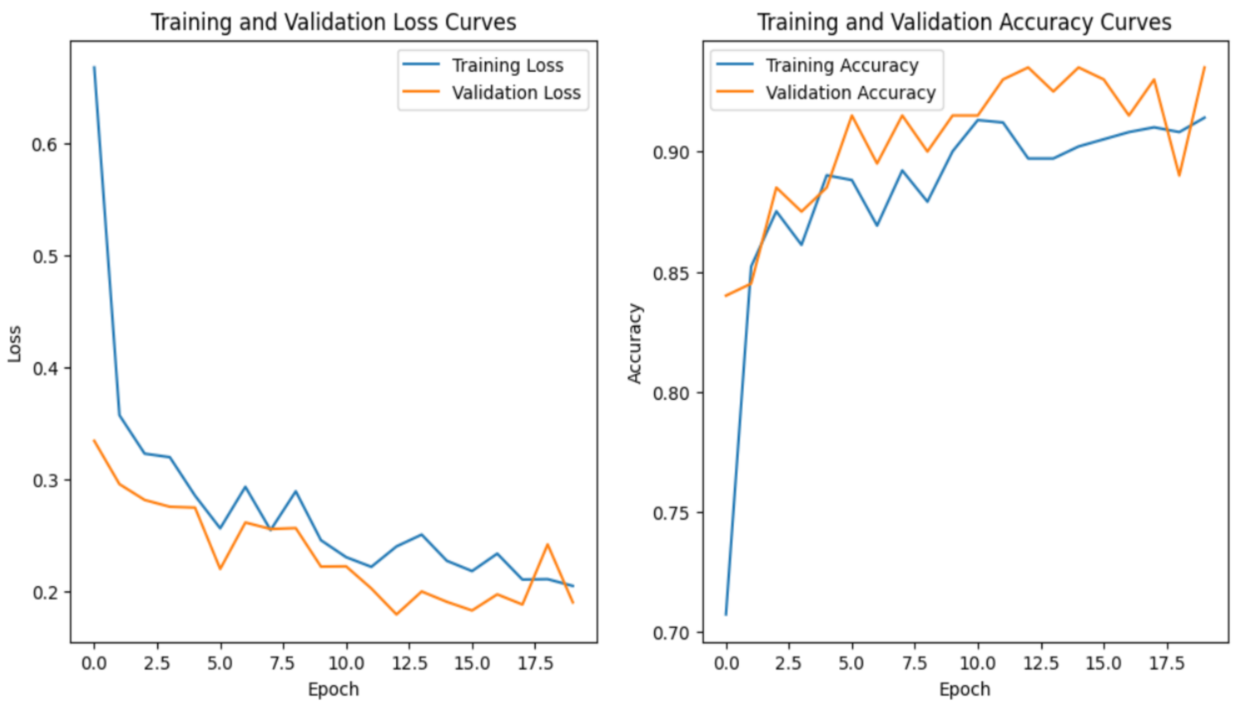


Fig 3.2 VGG16-based Brain Classifier

Many architecture were tested and the best output results were got by pre-trained model such as VGG16 and RESNET50 after comparing between both models with different architecture and adjusting each hyperparamters we found that using VGG16 with keeping the base layers, excluding the top layers (to customize the model), adding a dense layer of 256 and using regularization to reduce the overfitting by using Dropout() with percentage of 50% we got the best results.

3 – 3 Training Results

Training results were very acceptable after 20 epochs and the total training time for the model was nearly ~300s, results were as the following:

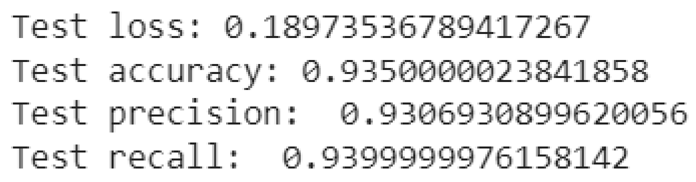


```
Train loss: 0.16949103772640228
Train accuracy: 0.9330669045448303
Train precision:  0.9155470132827759
Train recall:  0.9539999961853027
```

Fig 3.3 VGG16-based Brain Classifier Training Results

The training loss is 0.169 and the training accuracy is 0.933, which indicates that the model is performing well on the training data. The training precision is 0.916 and recall is 0.954, which means that the model is able to correctly identify a high percentage of positive cases (tumors) while minimizing false positives.

3 – 4 Testing Metrics



```
Test loss: 0.18973536789417267
Test accuracy: 0.9350000023841858
Test precision:  0.9306930899620056
Test recall:  0.9399999976158142
```

Fig 3.4 VGG16-based Brain Classifier Testing Results

As we can see the model scores very high on accuracy, but since this is a classification problem we should also pay attention to precision and recall. The model has both high precision and recall meaning that it has a very good distinction between the two classes "No Tumor" and "Tumor" in our dataset.

The test loss is 0.190 and the test accuracy is 0.935, which indicates that the model is performing well on the testing data as well. The test precision is 0.931 and recall is 0.940, which means that the model is able to generalize well to new data and is not overfitting to the training data.

3 – 5 Testing Results

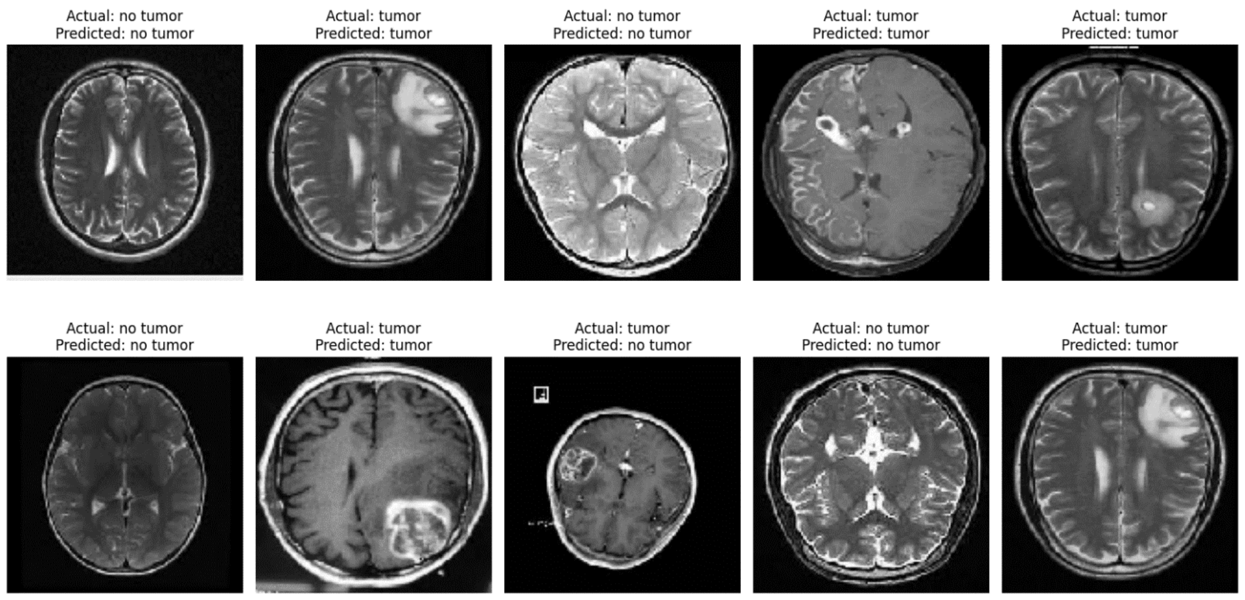


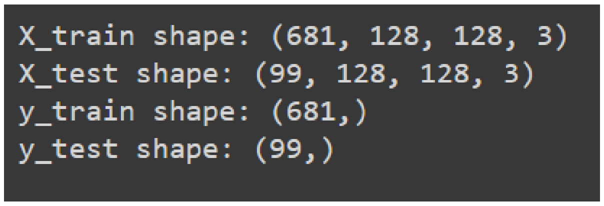
Fig 3.5 VGG16-based Brain Classifier Model Results

4 – Breast Tumor 2nd Stage Classification

4 – 1 Data Preparation

The data in first contained in breast scan folder as benign, malignant and normal each folder of these has a Train and Test folder.

The Train and Test folders contain the original and mask images so we work on the data to take the images without masks and append them in an x_train , y_train , x_test , y_test and each of y_train and y_test we append (0) for normal and (1) for malignant and (2) for benign and we resize the images to be (128,128) and then convert all lists to be array to work on , then we shuffle the data and then encode our data .



```
X_train shape: (681, 128, 128, 3)
X_test shape: (99, 128, 128, 3)
y_train shape: (681,)
y_test shape: (99,)
```

Fig 4.1 Train and Test Data Shapes

Then some data augmentation was performed.

Below is a visual representing the data that the classifier will work with:

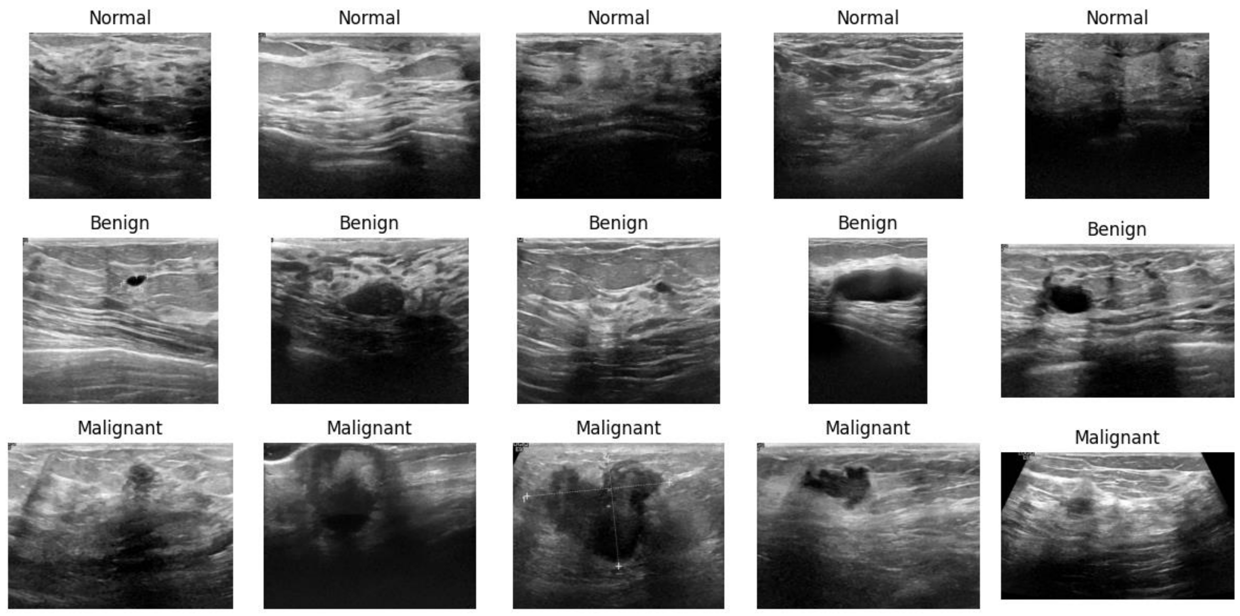


Fig 4.2 Breast Tumor Dataset Sample

4 – 2 Proposed Architecture

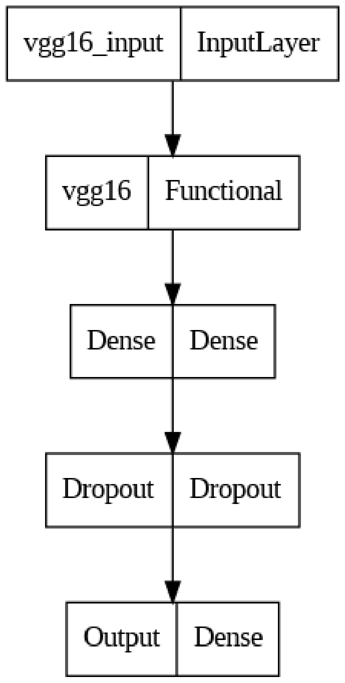


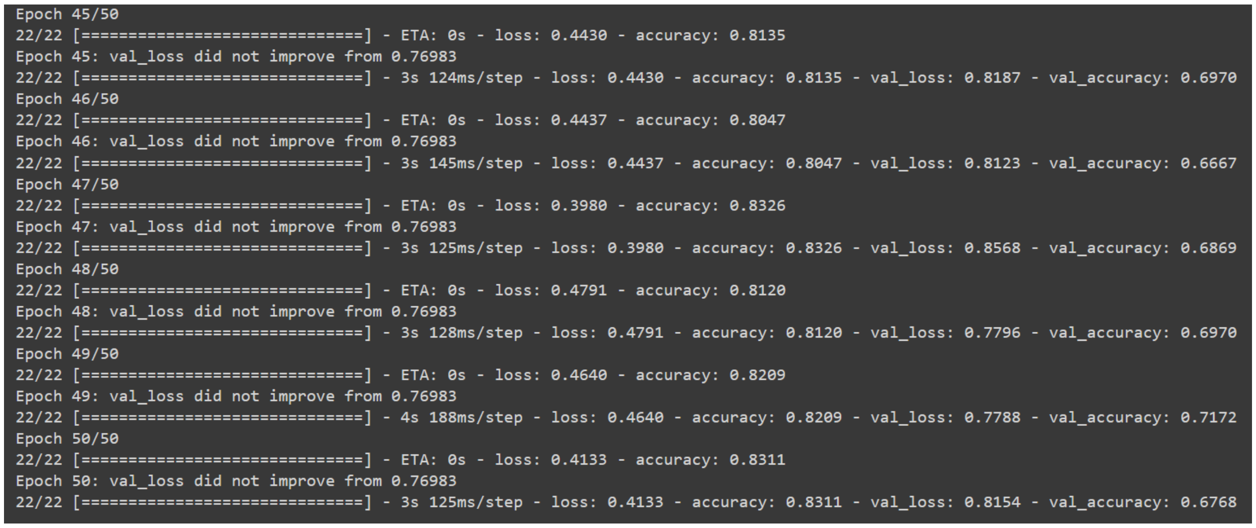
Fig 4.3 VGG16-based Breast Tumor Classifier

On top of the VGG16 model, we add two fully connected layers with ReLU activation and a softmax output layer with three units for classification into three classes: normal, benign, and malignant. We use the Dropout layer to prevent overfitting.

The model was then compiled with “Categorical Cross Entropy” as a loss function, with “Adam” optimizer.

4 – 3 Training Results

The model was trained for about 50 epochs,and it took about 200s to train. Below are the results of the training of the model:



```
Epoch 45/50
22/22 [==============================] - ETA: 0s - loss: 0.4430 - accuracy: 0.8135
Epoch 45: val_loss did not improve from 0.76983
22/22 [==============================] - 3s 124ms/step - loss: 0.4430 - accuracy: 0.8135 - val_loss: 0.8187 - val_accuracy: 0.6970
Epoch 46/50
22/22 [==============================] - ETA: 0s - loss: 0.4437 - accuracy: 0.8047
Epoch 46: val_loss did not improve from 0.76983
22/22 [==============================] - 3s 145ms/step - loss: 0.4437 - accuracy: 0.8047 - val_loss: 0.8123 - val_accuracy: 0.6667
Epoch 47/50
22/22 [==============================] - ETA: 0s - loss: 0.3980 - accuracy: 0.8326
Epoch 47: val_loss did not improve from 0.76983
22/22 [==============================] - 3s 125ms/step - loss: 0.3980 - accuracy: 0.8326 - val_loss: 0.8568 - val_accuracy: 0.6869
Epoch 48/50
22/22 [==============================] - ETA: 0s - loss: 0.4791 - accuracy: 0.8120
Epoch 48: val_loss did not improve from 0.76983
22/22 [==============================] - 3s 128ms/step - loss: 0.4791 - accuracy: 0.8120 - val_loss: 0.7796 - val_accuracy: 0.6970
Epoch 49/50
22/22 [==============================] - ETA: 0s - loss: 0.4640 - accuracy: 0.8209
Epoch 49: val_loss did not improve from 0.76983
22/22 [==============================] - 4s 188ms/step - loss: 0.4640 - accuracy: 0.8209 - val_loss: 0.7788 - val_accuracy: 0.7172
Epoch 50/50
22/22 [==============================] - ETA: 0s - loss: 0.4133 - accuracy: 0.8311
Epoch 50: val_loss did not improve from 0.76983
22/22 [==============================] - 3s 125ms/step - loss: 0.4133 - accuracy: 0.8311 - val_loss: 0.8154 - val_accuracy: 0.6768
```

Fig 4.4 VGG16-Based Breast Tumor Classifier Training Results (1)

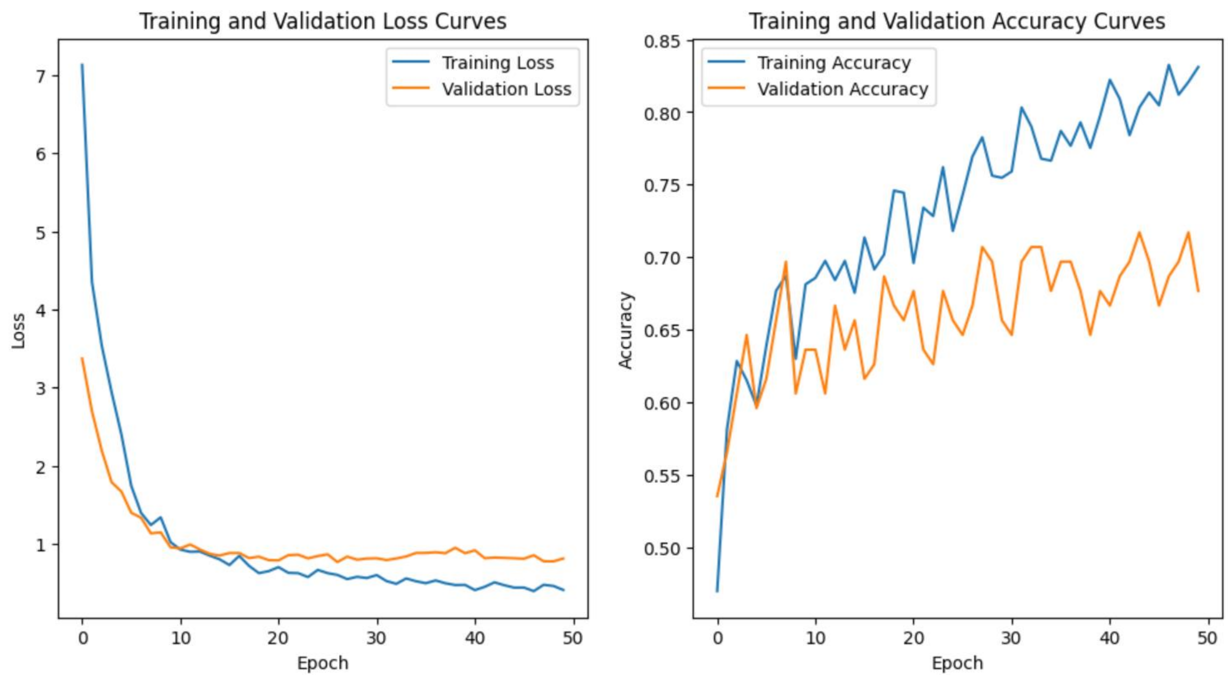
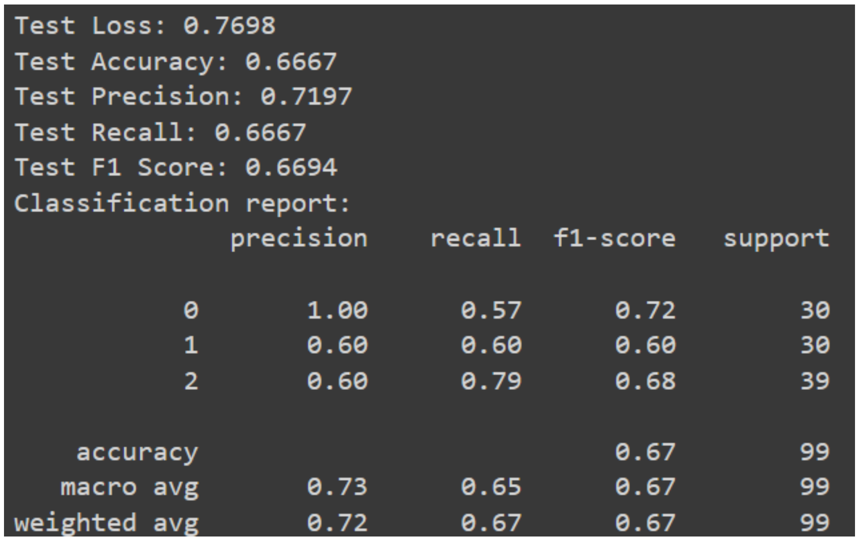


Fig 4.5 VGG16-Based Breast Tumor Classifier Training Results (2)

4 – 4 Testing Metrics



```
Test Loss: 0.7698
Test Accuracy: 0.6667
Test Precision: 0.7197
Test Recall: 0.6667
Test F1 Score: 0.6694
Classification report:
              precision    recall  f1-score   support

           0       1.00      0.57      0.72        30
           1       0.60      0.60      0.60        30
           2       0.60      0.79      0.68        39

    accuracy                           0.67        99
   macro avg       0.73      0.65      0.67        99
weighted avg       0.72      0.67      0.67        99
```

Fig 4.6 VGG16-Based Breast Tumor Classifier Classification Report

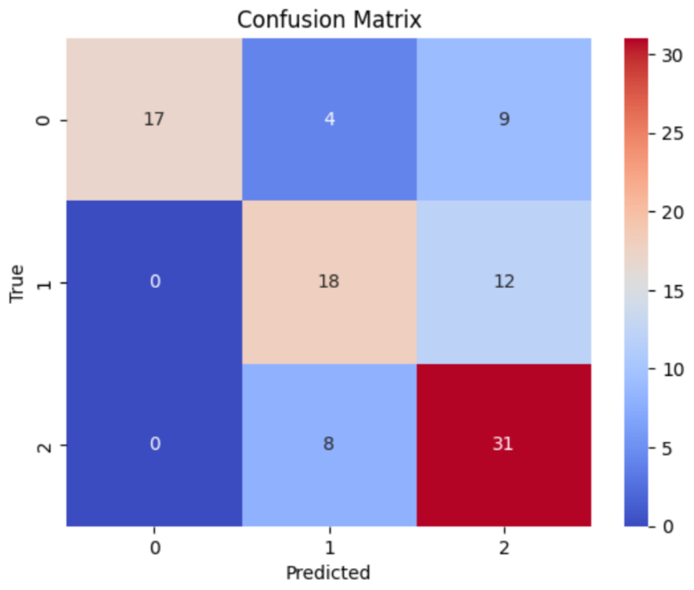


Fig 4.7 VGG16-Based Breast Tumor Classifier Confusion Matrix

The test loss is 0.7698 and the test accuracy is 0.6667, which indicates that the model is not performing well on the testing data. The test precision is 0.7197 and recall is 0.6667, which means that the model is able to correctly identify a moderate percentage of positive cases while minimizing false positives, but is not performing well overall.

4 – 5 Testing Results

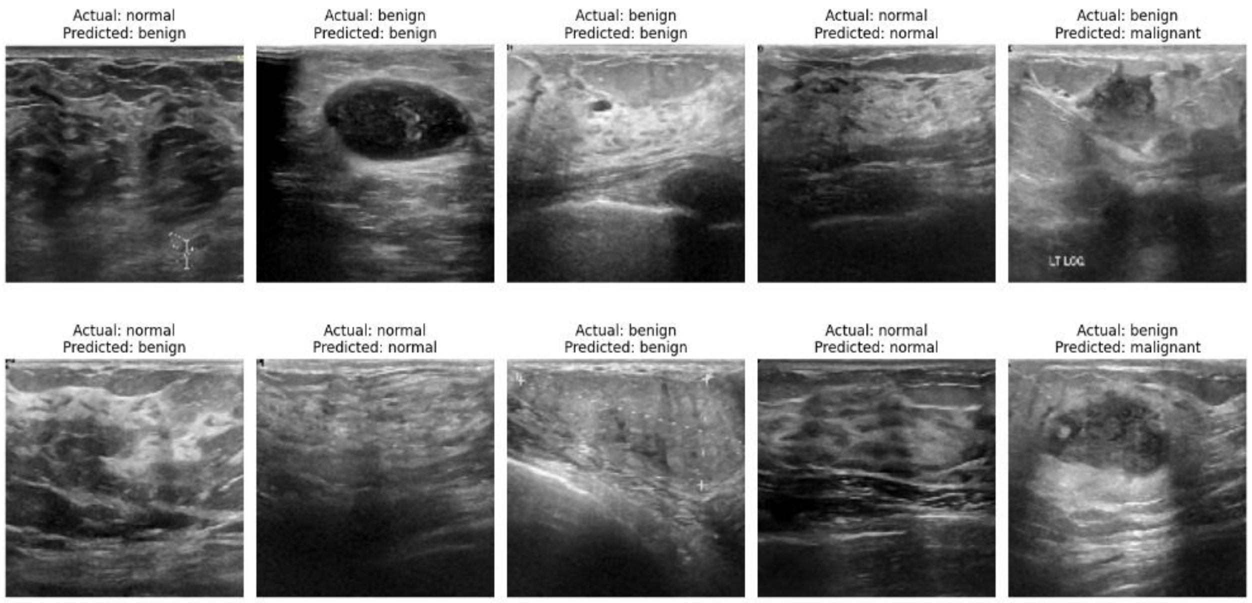


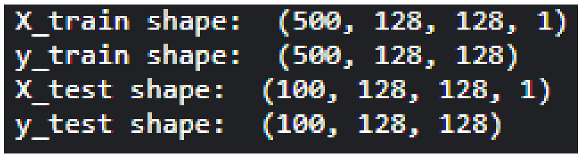
Fig 3.8 VGG16-Based Breast Tumor Classifier Results

Overall this model cannot be used reliably, we attribute this to having very few training samples to be able to predict the 3 classes, so the model theoretically should be able to perform better if more data was available for training.

5 – Brain Tumor Segmentation

5 – 1 Data Preparation

After classifying if the scan has a tumor on not, it is important to segment the tumor if available. The data of the brain tumors are in the form of a brain scan, and for each scan with a tumor there is a corresponding mask. The scans and brains are of various sizes, to unify this all the images were resized to be 128x128 pixels. Furthermore, the images were loaded as a single channel grey scale image. So, we have four arrays to work with X_train, y_train, X_test and y_test the shapes of the arrays are listed below:



```
X_train shape:  (500, 128, 128, 1)
y_train shape:  (500, 128, 128)
X_test shape:  (100, 128, 128, 1)
y_test shape:  (100, 128, 128)
```

Fig 5.1 Train and Test Data Shapes

As can be we have 500 samples for training and about 100 samples for testing, both training and testing sets are loaded with shape 128x128 pixels and as a single channel greyscale image. Below is a visualization that demonstrates how the data looks like:

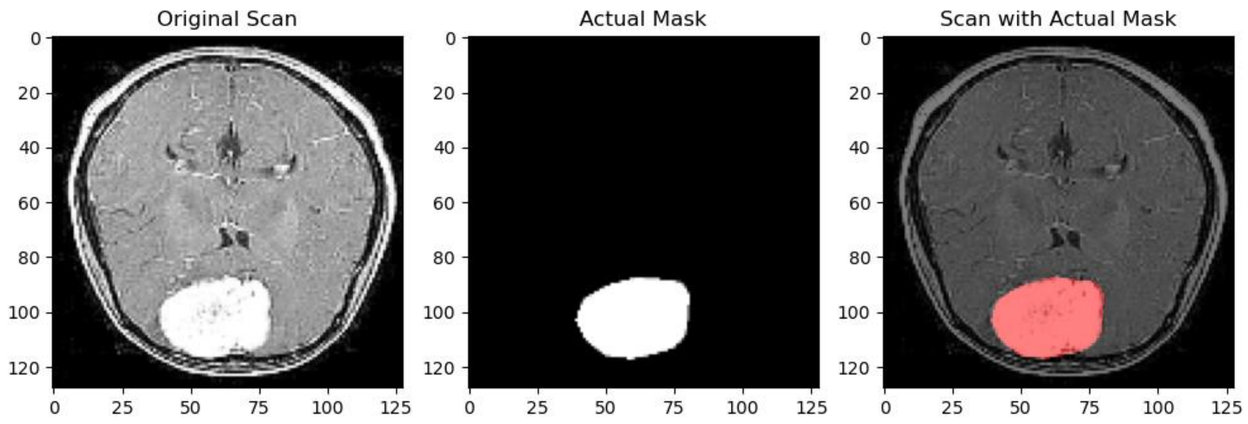


Fig 5.2 Brain Tumor Dataset Sample

5 – 2 Proposed Architecture

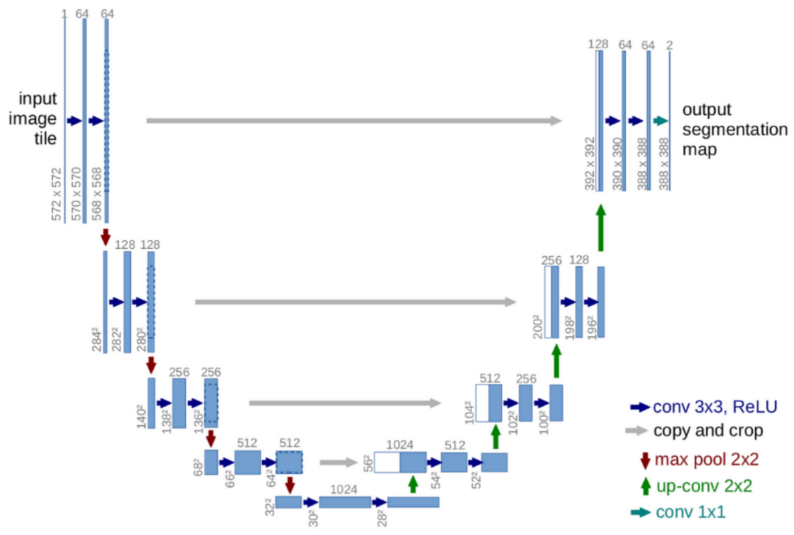


Fig 5.3 U-NET Architecture

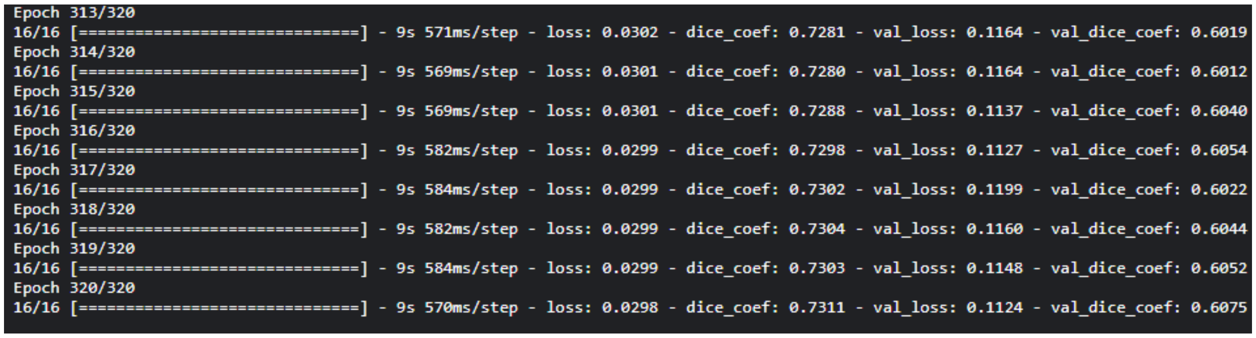
This model is an implementation of the U-Net architecture, which is a popular deep learning model for image segmentation tasks. The architecture consists of an encoder-decoder network with skip connections that allow the model to preserve spatial information and capture both local and global features of the input image.

The input to the model is an image tensor, and the output is a binary mask indicating the segmentation of the input image. The architecture starts with a series of convolutional layers that progressively downsample the input image, followed by a bottleneck layer that captures the most important features of the input. The upsampling path then uses transposed convolutional layers to upsample the bottleneck features and concatenate them with the corresponding features from the downsampling path to recover the spatial information. The final layer uses a sigmoid activation function to output a binary mask that indicates the presence or absence of the segmented object.

We used the original implementation as the UNET original paper, the only difference being we introduced Dropout layers as regularization in order to prevent overfitting.

5 – 3 Training Results

The model was trained for about 320 epochs, using binary cross entropy as a loss function, and Adam optimization with learning rate = 0.0001, and it took about 45 minutes to train, we also used dice coefficient as our metric to decide the fitness of the model, below are the results of training:



```
Epoch 313/320
16/16 [==============================] - 9s 571ms/step - loss: 0.0302 - dice_coef: 0.7281 - val_loss: 0.1164 - val_dice_coef: 0.6019
Epoch 314/320
16/16 [==============================] - 9s 569ms/step - loss: 0.0301 - dice_coef: 0.7280 - val_loss: 0.1164 - val_dice_coef: 0.6012
Epoch 315/320
16/16 [==============================] - 9s 569ms/step - loss: 0.0301 - dice_coef: 0.7288 - val_loss: 0.1137 - val_dice_coef: 0.6040
Epoch 316/320
16/16 [==============================] - 9s 582ms/step - loss: 0.0299 - dice_coef: 0.7298 - val_loss: 0.1127 - val_dice_coef: 0.6054
Epoch 317/320
16/16 [==============================] - 9s 584ms/step - loss: 0.0299 - dice_coef: 0.7302 - val_loss: 0.1199 - val_dice_coef: 0.6022
Epoch 318/320
16/16 [==============================] - 9s 582ms/step - loss: 0.0299 - dice_coef: 0.7304 - val_loss: 0.1160 - val_dice_coef: 0.6044
Epoch 319/320
16/16 [==============================] - 9s 584ms/step - loss: 0.0299 - dice_coef: 0.7303 - val_loss: 0.1148 - val_dice_coef: 0.6052
Epoch 320/320
16/16 [==============================] - 9s 570ms/step - loss: 0.0298 - dice_coef: 0.7311 - val_loss: 0.1124 - val_dice_coef: 0.6075
```

Fig 4.4 U-NET Brain Segmentation Training Results (1)

And the first 100 epochs of training:

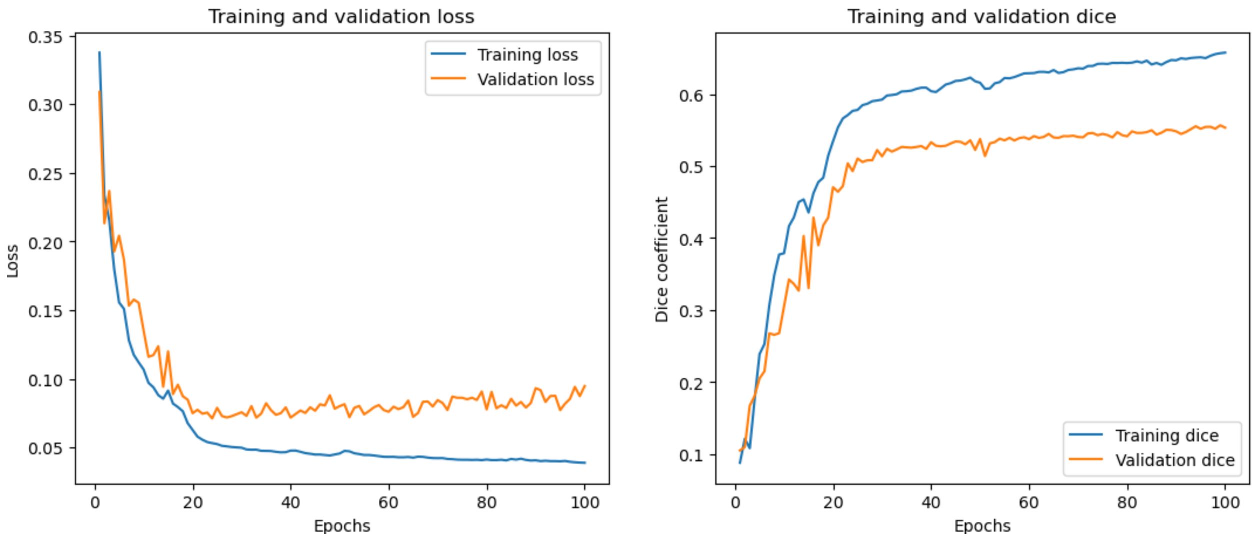
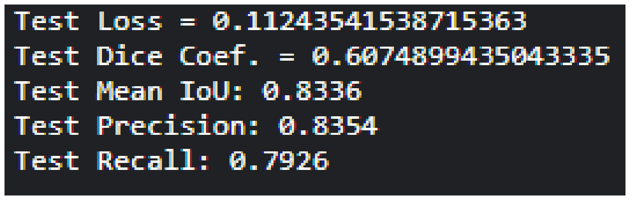


Fig 4.5 U-NET Brain Segmentation Training Results (2)

5 – 4 Testing Metrics & Results

Below are the results of testing the model on our test set:



```
Test Loss = 0.11243541538715363
Test Dice Coef. = 0.6074899435043335
Test Mean IoU: 0.8336
Test Precision: 0.8354
Test Recall: 0.7926
```

Fig 4.6 U-NET Brain Segmentation Test Metrics

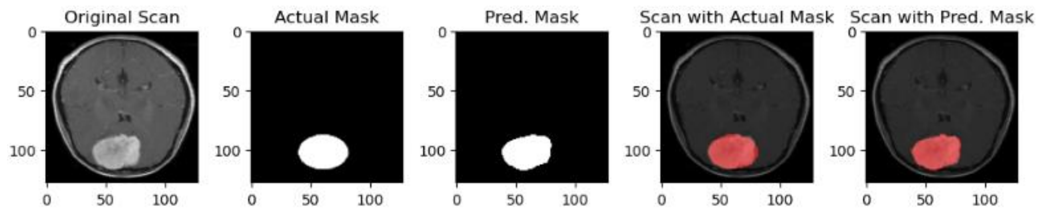
We can say that we have obtained a good fit. A higher MeanIOU value indicates that the predicted segmentation is more accurate and closer to the ground truth segmentation. Precision is a measure of how well the model identifies the true positive pixels (i.e., pixels that are part of the object of interest) among all the pixels it identified as positive.

High precision indicates that the model accurately identified the pixels belonging to the object of interest and didn't include too many false positive pixels (i.e., pixels that are not part of the object of interest).

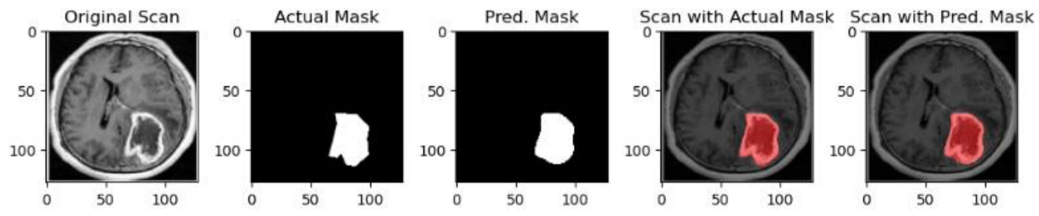
Recall, on the other hand, measures the ability of the model to identify all positive pixels (i.e., both true positive and false negative pixels). High recall indicates that the model is able to capture most of the object of interest, and did not miss too many true positive pixels.

5 – 5 Testing Results

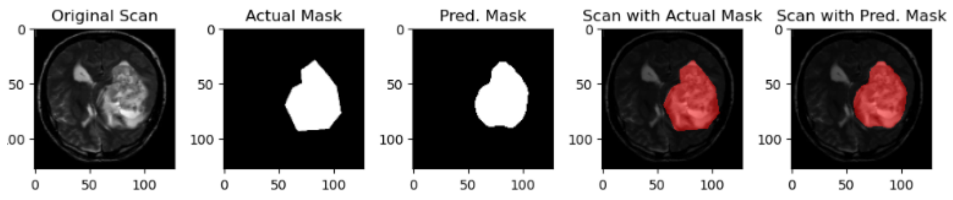
Below are samples of testing the model on test images with the segmented tumor width and height:



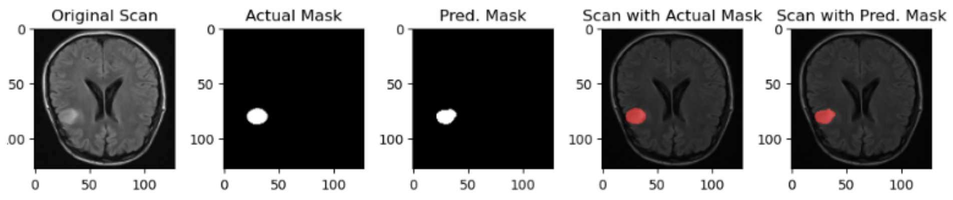
```
Predicted Tumor width in pixels : 42
Predicted Tumor height in pixels : 30
```



```
Predicted Tumor width in pixels : 33
Predicted Tumor height in pixels : 43
```



```
Predicted Tumor width in pixels : 49
Predicted Tumor height in pixels : 60
```



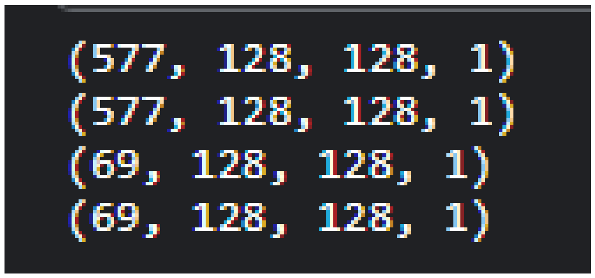
```
Predicted Tumor width in pixels : 18
Predicted Tumor height in pixels : 14
```

Fig 5.7 Brain Segmentation Model Results

6 –Breast Tumor Segmentation

6 – 1 Data Preparation

The data of the breast tumor are in the form of a breast ultrasound images, and for each image with a tumor there is a corresponding mask. The images and mask are of various sizes, to unify this all the images were resized to be 128x128 pixels. Furthermore, the images were loaded as a single channel grey scale image. So, we have four arrays to work with X_train, y_train, X_test and y_test the shapes of the arrays are listed below:



```
(577, 128, 128, 1)
(577, 128, 128, 1)
(69, 128, 128, 1)
(69, 128, 128, 1)
```

Fig 6.1 Train and Test Data Shapes

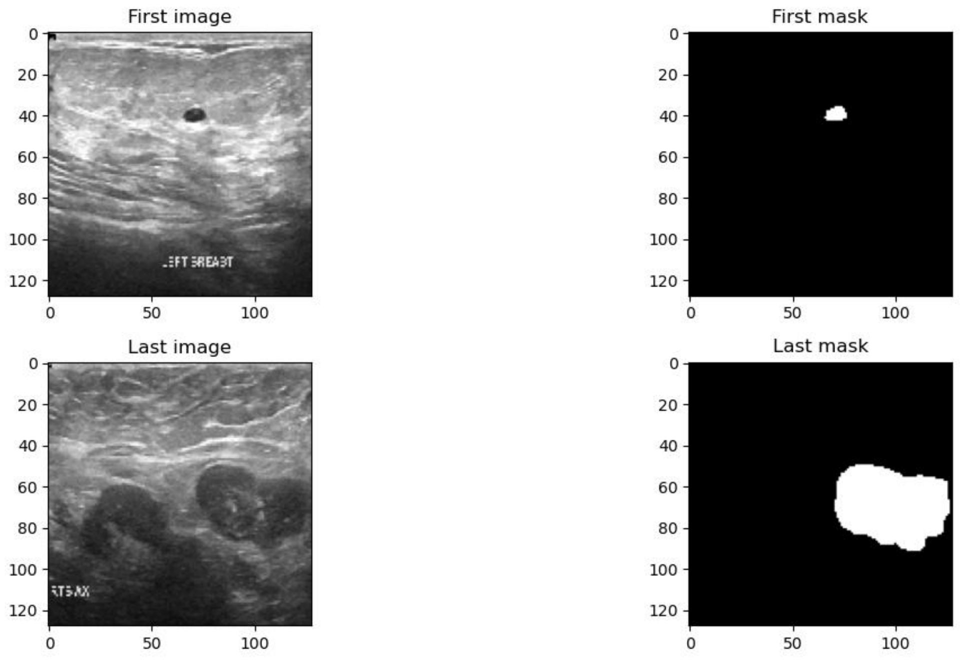


Fig 6.2 Breast Tumor Dataset Sample

6 – 2 Proposed Architecture

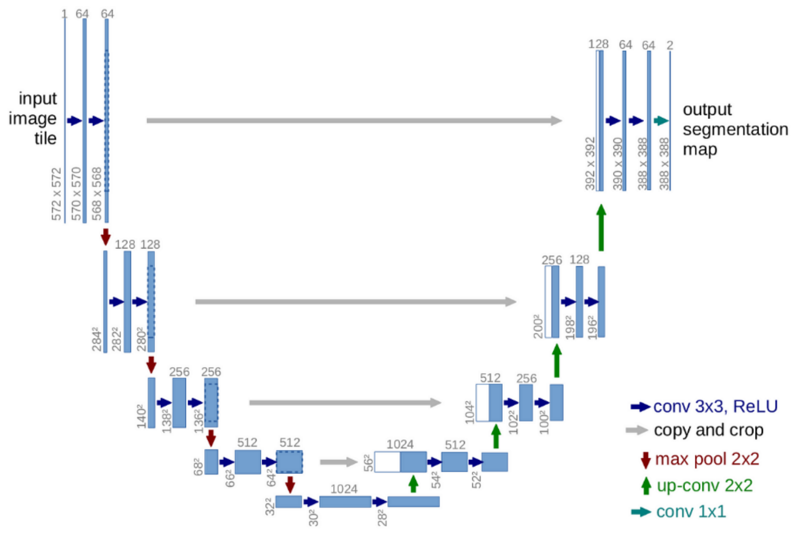


Fig 6.3 U-NET Architecture

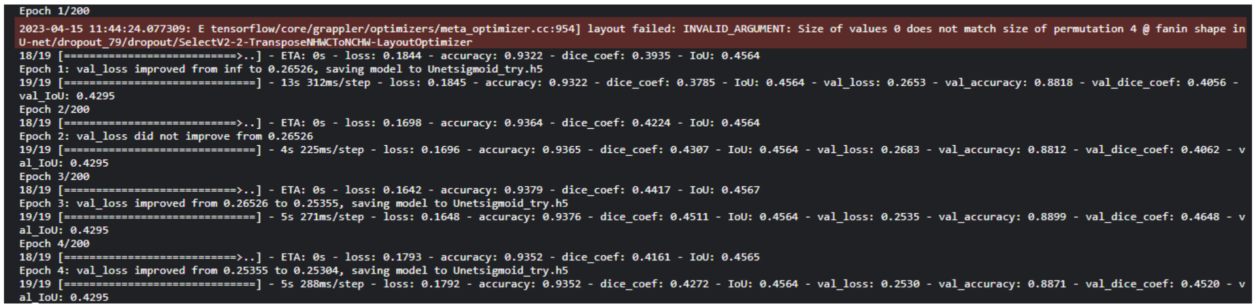
The input to the U-Net is a 2D image with a specified input size (128,128,1). The contracting path involves four sets of convolutional layers followed by max pooling layers to reduce the spatial dimensions of the feature maps while increasing the number of channels. Each set consists of two convolutional layers with a ReLU activation function and a dropout layer for regularization. The number of filters in each convolutional layer increases as we move deeper into the network.

After the contracting path, the expanding path is symmetric to the contracting path and consists of four sets of convolutional layers followed by transpose convolutional layers to upsample the feature maps while reducing the number of channels. Each set consists of two convolutional layers with a ReLU activation function and a dropout layer for regularization. The number of filters in each convolutional layer decreases as we move deeper into the network.

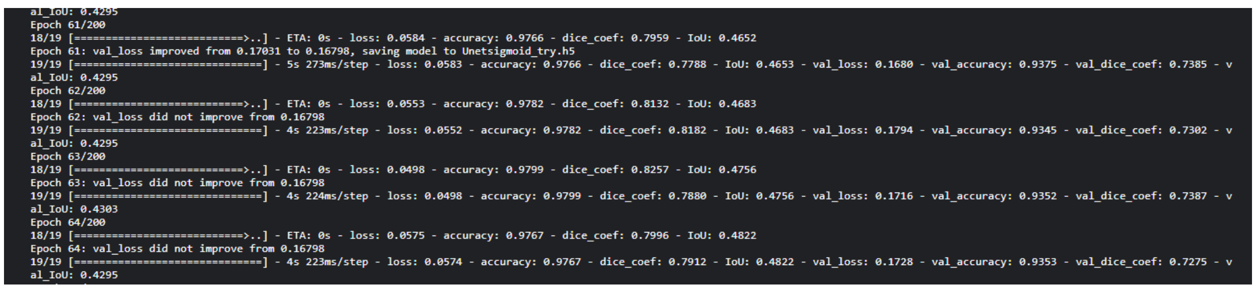
The expanding path also includes concatenation of the feature maps from the corresponding contracting path layers to preserve the high-resolution information. Finally, a 1x1 convolutional layer with a sigmoid activation function is applied to generate the output segmentation map.

6 – 3 Training Results

The model was trained for about 91 epochs, using binary cross entropy as a loss function, and Adam optimization with learning rate = 0.0005, we also used dice coefficient, accuracy and as our metric to decide the fitness of the model. The model took 15 minutes to train. Below are the results of training:



```
Epoch 1/200
2023-04-15 11:44:24.077309: E tensorflow/core/grappler/optimizers/meta_optimizer.cc:954] layout failed: INVALID_ARGUMENT: Size of values 0 does not match size of permutation 4 @ fanin shape in
U-net/dropout_79/dropout/SelectV2-2-TransposeNHWCToNCHW-LayoutOptimizer
18/19 [===========================>..] - ETA: 0s - loss: 0.1844 - accuracy: 0.9322 - dice_coef: 0.3935 - IoU: 0.4564
Epoch 1: val_loss improved from inf to 0.26526, saving model to Unetsigmoid_try.h5
19/19 [==============================] - 13s 312ms/step - loss: 0.1845 - accuracy: 0.9322 - dice_coef: 0.3785 - IoU: 0.4564 - val_loss: 0.2653 - val_accuracy: 0.8818 - val_dice_coef: 0.4056 -
val_IoU: 0.4295
Epoch 2/200
18/19 [===========================>..] - ETA: 0s - loss: 0.1698 - accuracy: 0.9364 - dice_coef: 0.4224 - IoU: 0.4564
Epoch 2: val_loss did not improve from 0.26526
19/19 [==============================] - 4s 225ms/step - loss: 0.1696 - accuracy: 0.9365 - dice_coef: 0.4307 - IoU: 0.4564 - val_loss: 0.2683 - val_accuracy: 0.8812 - val_dice_coef: 0.4062 - v
al_IoU: 0.4295
Epoch 3/200
18/19 [===========================>..] - ETA: 0s - loss: 0.1642 - accuracy: 0.9379 - dice_coef: 0.4417 - IoU: 0.4567
Epoch 3: val_loss improved from 0.26526 to 0.25355, saving model to Unetsigmoid_try.h5
19/19 [==============================] - 5s 271ms/step - loss: 0.1648 - accuracy: 0.9376 - dice_coef: 0.4511 - IoU: 0.4564 - val_loss: 0.2535 - val_accuracy: 0.8899 - val_dice_coef: 0.4648 - v
al_IoU: 0.4295
Epoch 4/200
18/19 [===========================>..] - ETA: 0s - loss: 0.1793 - accuracy: 0.9352 - dice_coef: 0.4161 - IoU: 0.4565
Epoch 4: val_loss improved from 0.25355 to 0.25304, saving model to Unetsigmoid_try.h5
19/19 [==============================] - 5s 288ms/step - loss: 0.1792 - accuracy: 0.9352 - dice_coef: 0.4272 - IoU: 0.4564 - val_loss: 0.2530 - val_accuracy: 0.8871 - val_dice_coef: 0.4520 - v
al_IoU: 0.4295
```



```
al_IoU: 0.4295
Epoch 61/200
18/19 [===========================>..] - ETA: 0s - loss: 0.0584 - accuracy: 0.9766 - dice_coef: 0.7959 - IoU: 0.4652
Epoch 61: val_loss improved from 0.17031 to 0.16798, saving model to Unetsigmoid_try.h5
19/19 [==============================] - 5s 273ms/step - loss: 0.0583 - accuracy: 0.9766 - dice_coef: 0.7788 - IoU: 0.4653 - val_loss: 0.1680 - val_accuracy: 0.9375 - val_dice_coef: 0.7385 - v
al_IoU: 0.4295
Epoch 62/200
18/19 [===========================>..] - ETA: 0s - loss: 0.0553 - accuracy: 0.9782 - dice_coef: 0.8132 - IoU: 0.4683
Epoch 62: val_loss did not improve from 0.16798
19/19 [==============================] - 4s 223ms/step - loss: 0.0552 - accuracy: 0.9782 - dice_coef: 0.8182 - IoU: 0.4683 - val_loss: 0.1794 - val_accuracy: 0.9345 - val_dice_coef: 0.7302 - v
al_IoU: 0.4295
Epoch 63/200
18/19 [===========================>..] - ETA: 0s - loss: 0.0498 - accuracy: 0.9799 - dice_coef: 0.8257 - IoU: 0.4756
Epoch 63: val_loss did not improve from 0.16798
19/19 [==============================] - 4s 224ms/step - loss: 0.0498 - accuracy: 0.9799 - dice_coef: 0.7880 - IoU: 0.4756 - val_loss: 0.1716 - val_accuracy: 0.9352 - val_dice_coef: 0.7387 - v
al_IoU: 0.4303
Epoch 64/200
18/19 [===========================>..] - ETA: 0s - loss: 0.0575 - accuracy: 0.9767 - dice_coef: 0.7996 - IoU: 0.4822
Epoch 64: val_loss did not improve from 0.16798
19/19 [==============================] - 4s 223ms/step - loss: 0.0574 - accuracy: 0.9767 - dice_coef: 0.7912 - IoU: 0.4822 - val_loss: 0.1728 - val_accuracy: 0.9353 - val_dice_coef: 0.7275 - v
al_IoU: 0.4295
```

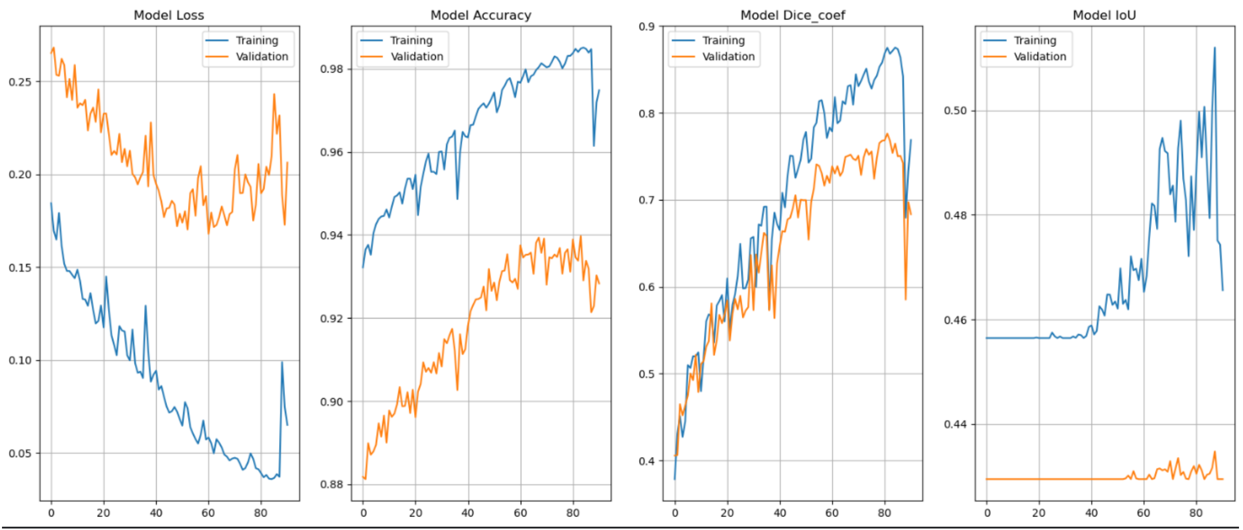
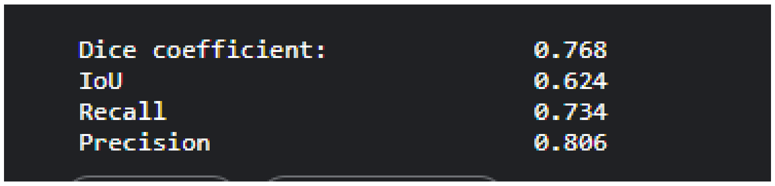


Fig 6.4 U-NET Breast Segmentation Training Results

6 – 4 Testing Metrics

Below are the results of testing the model on our test set:



```
Dice coefficient:           0.768
IoU                         0.624
Recall                      0.734
Precision                   0.806
```

Fig 6.5 U-NET Breast Segmentation Test Metrics

The Dice coefficient is a measure of the overlap between the predicted and ground truth segmentation masks, where a value of 1 indicates perfect overlap and 0 indicates no overlap. In this case, the Dice coefficient is 0.768, which means that the predicted segmentation mask has a high degree of overlap with the ground truth mask.

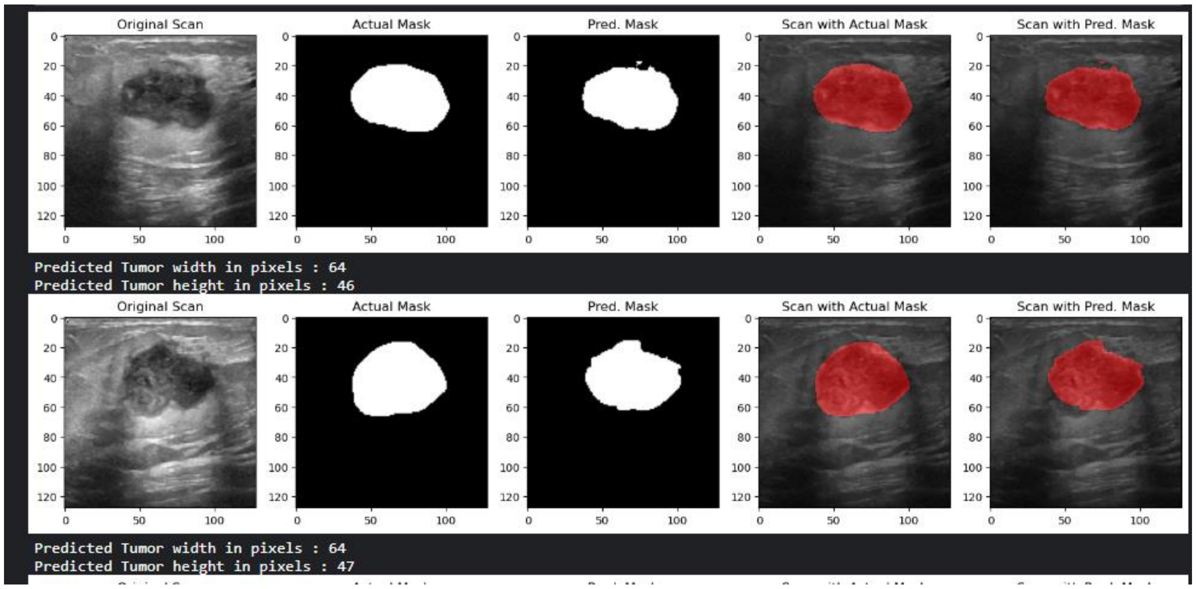
The IoU is another measure of the overlap between the predicted and ground truth masks, where a value of 1 indicates perfect overlap and 0 indicates no overlap. In this case, the IoU is 0.624, which means that the predicted mask covers 62.4% of the area of the ground truth mask.

The recall is a measure of the percentage of true positive pixels that were correctly identified by the model, while the precision is a measure of the percentage of predicted positive pixels that were actually true positive. In this case, the recall is 0.734, which means that the model correctly identified 73.4% of the true positive pixels, while the precision is 0.806, which means that 80.6% of the predicted positive pixels were actually true positive.

Overall, these results suggest that the model is performing well on the semantic segmentation task, with a high Dice coefficient, moderate IoU, and high recall and precision. However, it's important to consider the specific context of the segmentation task and the specific dataset being used to evaluate performance. These metrics may not always be appropriate or sufficient for evaluating the performance of a segmentation model, and additional metrics and visual inspection may be necessary to ensure that the model is performing well in practice.

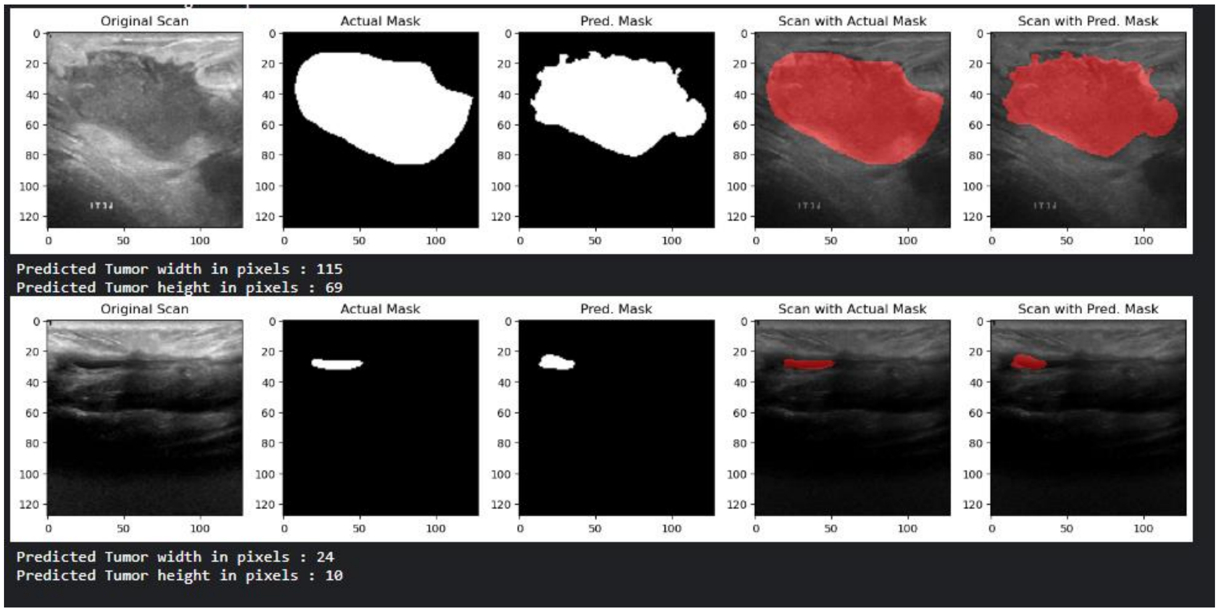
6 – 5 Testing Results

Below are samples of testing the model on test images with the segmented tumor width and height:



```
Predicted Tumor width in pixels : 64
Predicted Tumor height in pixels : 46
```

```
Predicted Tumor width in pixels : 64
Predicted Tumor height in pixels : 47
```



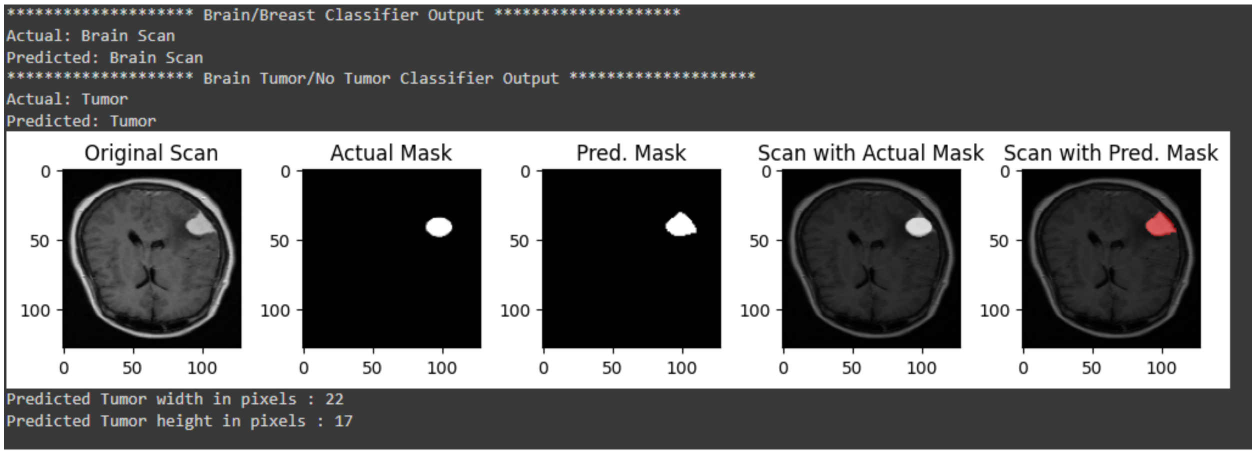
```
Predicted Tumor width in pixels : 115
Predicted Tumor height in pixels : 69
```

```
Predicted Tumor width in pixels : 24
Predicted Tumor height in pixels : 10
```

Fig 6.6 Breast Segmentation Model Results

7 – System Simulation

We will start the simulation by testing the brain classifier and segmentation, we will load a tumor image and feed it to the brain/breast scan classifier, then we will feed it to the brain tumor model, then we will feed it to the brain tumor segmentation model, and here are the results:

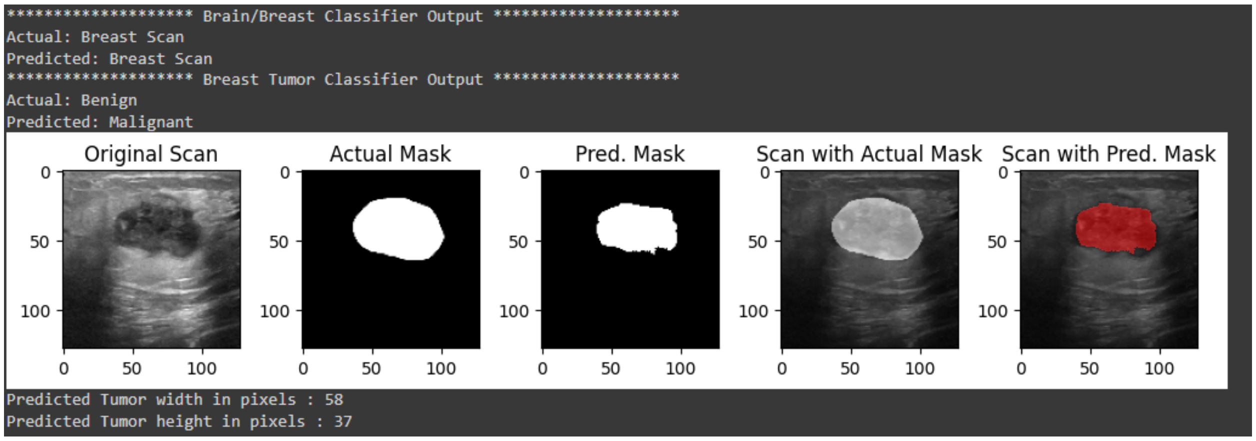


```
******************** Brain/Breast Classifier Output ********************
Actual: Brain Scan
Predicted: Brain Scan
******************** Brain Tumor/No Tumor Classifier Output ********************
Actual: Tumor
Predicted: Tumor
```

```
Predicted Tumor width in pixels : 22
Predicted Tumor height in pixels : 17
```

Fig 7.1 Brain Scans System Simulation

Similarly, we will test the breast classifier and segmentation model on a brain scan image that has a benign tumor in it, and here are the results:



```
******************** Brain/Breast Classifier Output ********************
Actual: Breast Scan
Predicted: Breast Scan
******************** Breast Tumor Classifier Output ********************
Actual: Benign
Predicted: Malignant
```

```
Predicted Tumor width in pixels : 58
Predicted Tumor height in pixels : 37
```

Fig 7.2 Brain Scans System Simulation