Assignment 3: Image Segmentation

Soham Padhye

March 19, 2024

1 Introduction

In this assignment, the objective is to train a segmentation model using a MobileNet pre-trained on the ImageNet dataset as an encoder and design a decoder that predicts segmented masks.

2 Dataset

ISIC 2016 dataset. This dataset has 900 training images, Segmented masks for training images. 379 test images, Segmented masks for test images.

3 Experimental setup

Used personal laptop with NVIDIA graphic card for model training. For training the 100 epochs it took around 4 for complete model training.

4 Code

This is the main code file where I have compiled all the results for 3 different network architecture.

Google colab code link

4.1 Code for augmented dataset

I have run this code separately as it is very heavy to run in single big file. You can refer this to get the results from Augmented dataset for model architecture segNet attention and atrous (explained further in report)

Google colab code link for augmentd dataset model

5 Methodology

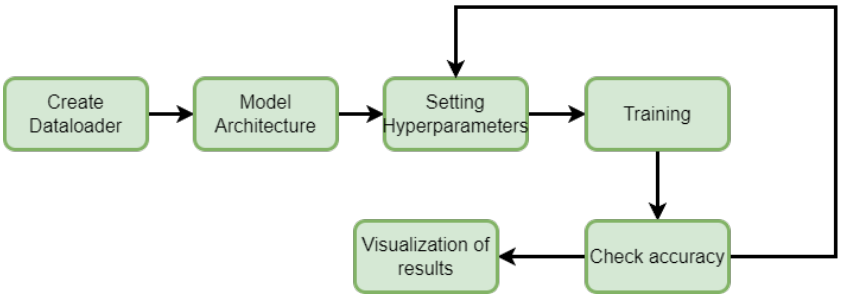


Figure 1: Methodology

6 Pre-Processing

6.1 Data Loader

1. Create the custom dataloader to load the images and corresponding masks.
2. Define transform to resize the image and convert it to tensor.

6.2 Augmentation

1. Apply augmentation technique for increasing the size of the dataset and making model to learn the good generalization.
2. Apply random crop, horizontal flip, rotate, color jitter and Gaussian noise transform as an augmentation. I kept original training images as it is and created new images with these transformations. Then combine original training examples with augmented exampled to create new big dataset.
3. Sample images loaded from the dataloader with augmentation are as shown below

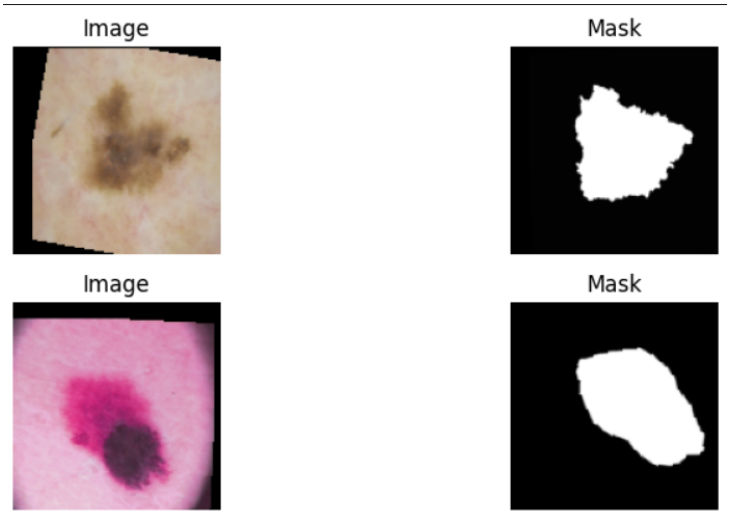


Figure 2: Training images and mask

7 Feature extraction

For this task use a pre-trained MobileNet encoder trained on ImageNetV2. Then design a custom decoder atop this encoder for the segmentation task. Train the decoder while keeping the encoder frozen.

7.1 Architecture 1- UNet with skip connections

- Initially I started with the architecture which is similar to **UNet**. Designed the decoder for increasing the size of image and concatenated the corresponding encoder output for getting better reconstruction of an image mask.
- In the decoder blocks I used 3 convolution layers followed by ReLU and finally concatenating this with encoder part.
- Used Binary cross entropy with logits loss for model tuning with learning rate of 0.001 for training.
- With this architecture I was able to achieve the **Mean IoU on testing data around 0.77 and Dice score 0.87**. In this test case some of part was False positive because of dilemma in input image.

7.2 Architecture 2- SegNet with attention gate !!

- In this architecture I used the concept of attention in the transformer.Concept is to create a gate to pass the features that are important for segmentation. This will help in creating the region of interest for finding the image mask.
- Created the attention gate with Convolution layer followed by sigmoid and applied over the encoder output features. This will give weightage to the output of encoder and then concatenated with decoder.
- SegNet is one of the state-of-the-art architecture for segmentation. In this architecture I concatenated encoder with corresponding decoder. Then perform the up-convolution apply ReLU. After this pass the tensor to 3 consecutive convolution layers followed by batch-normalization followed ReLU. This is same architecture as **SegNet** research paper.
- As similar to the architecture-2, I added attention gate on encoder output and then concatenated encoder and decoder.
- With this modification there is an increase in the mean IoU and dice score. Visual results was also good. It was able to detect the small regions also.

7.3 Architecture 3- SegNet with attention map and Atrous (Dilated) convolution !!!

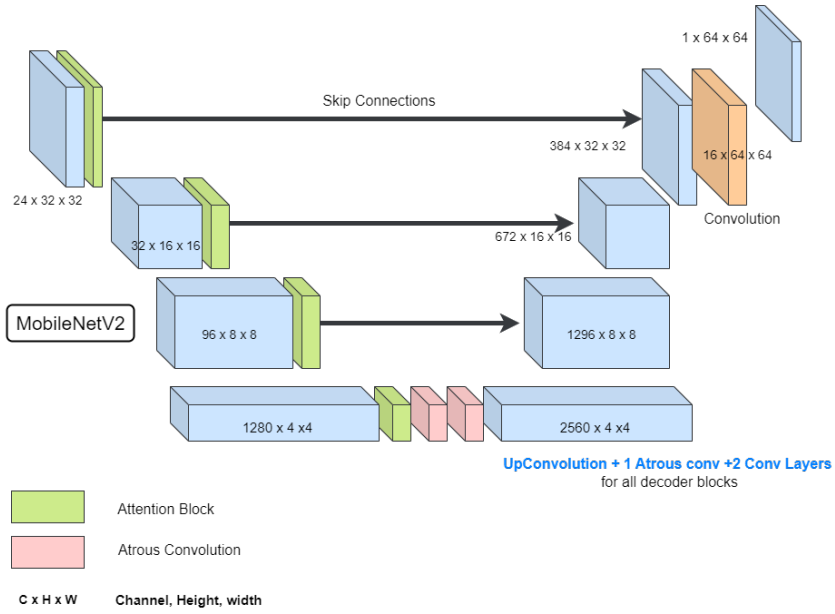


Figure 3: Network architecture

- **Atrous convolution** is concept coined by Google team in research paper for semantic segmentation. Atrous convolution, a powerful tool that allows us to explicitly control the resolution of features computed by deep convolutional neural networks and adjust filter's field-of-view in order to capture multi-scale information, generalizes standard convolution operation.

- With the use of Atrous convolution with dilation rate of 1 and 2 I extracted the features from encoder and then I passed these features to decoder. This will give us more semantics as it adjusts its field of view while performing convolution.

- Model performance is best for BCE logits loss with learning rate of 0.001 for 100 epochs. With this setup the segmentation model is able to achieve the very good **IoU 0.81 and Dice score 0.88 on testing dataset.**

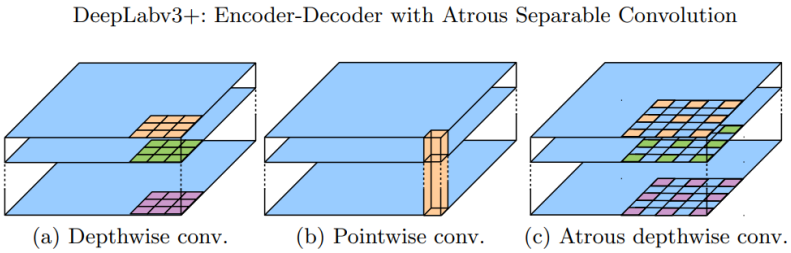


Figure 4: DeepLab Atrous Conv

- But good things always come with some drawbacks, and In my model architecture-3, the drawback is huge number of parameters and greater model size!!!

- As the performance of this model is fantastic I fixed this model and done further experiments.

8 Model Training

I trained the architecture 4- SegNet with attention map and Atrous (Dilated) convolution for various hyperparameters.

8.1 Loss function

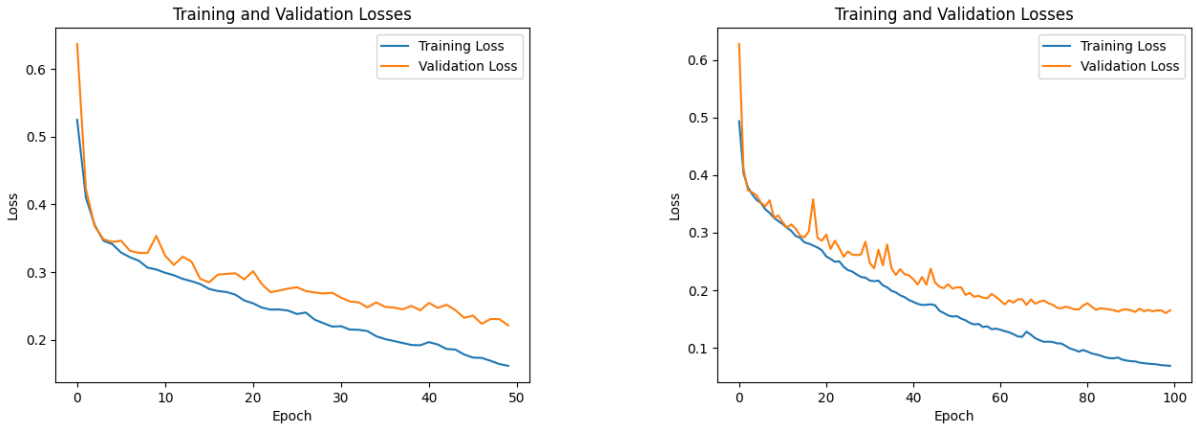
- Loss function plays an important role in the model optimization. I tried IoU loss function. In this loss function we calculate the IoU and then subtract it from 1 that will provide us the loss.
- Dice loss can also be used as loss function same as IoU loss. Then I tried the combination of Dice loss and IoU loss by providing some weightage to IoU and Dice loss. In some cases with this loss function I was getting negative values. So I haven't used these losses for my model.

8.1.1 Advantages of BCE with logits:

- The sigmoid activation and binary cross-entropy loss are combined into a single operation, improve computational efficiency.
- The logit transformation before applying the loss helps prevent numerical instability issues, especially when dealing with very small or very large values.
- It simplifies the process of implementing binary classification tasks, such as foreground-background segmentation, by providing a single loss function.

9 Plots and observations

9.1 Training and validation loss



(a) Training Loss for 50 epochs

(b) training Loss for 100 epochs

Figure 5: Training and Validation Loss

From the figure of loss versus the number of epochs for each cross validation set, it can be observed that both the training and validation losses decreases as the number of epochs increases. This kind of behavior indicates that the model is continuously improving over the training data and it has ability to generalize well to unseen validation data.

The decreasing validation loss suggests that the model is not only memorizing the training data but is also generalizing well to new unseen data. This property is needed for ensuring that the model to performs well on real-world unseen data.

But from the graph of loss vs no. of epochs it can be seen that after 50 epochs the model is not converged. There is still chance to decrease the loss further. So I trained the model again for 100 epochs. After this the model is converged.

9.2 IoU and Dice score on test dataset

Mean IoU after testing the model architecture 4 is around 0.80 and mean Dice score is 0.88. IoU values and Dice score for different model architectures are give in table below table.

Architecture	mean IoU	Dice Score
UNet	0.77	0.858
SegNet + attetion	0.58	0.70
SegNet + attetion + Atrous conv	**0.78**	**0.871**

Table 1: IoU and Dice score for different architectures

The results on test images on SegNet + attetion + Atrous conv this model is shown in figure 7.

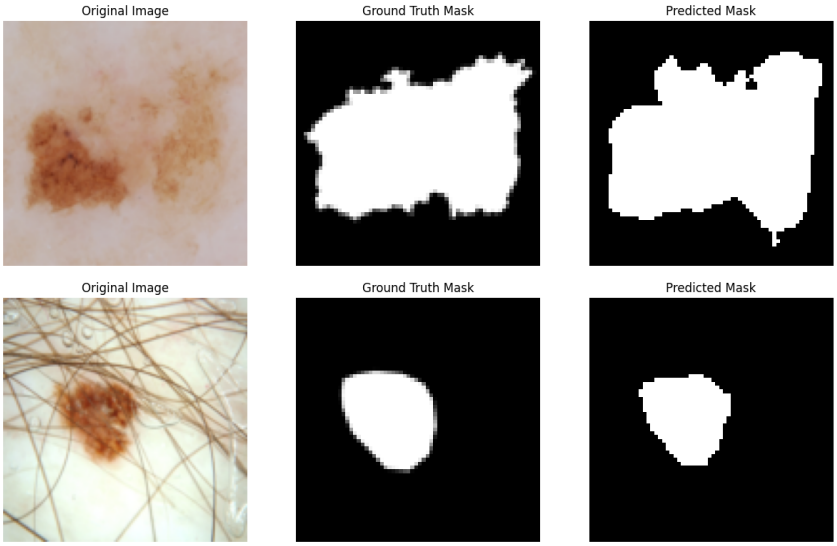


Figure 6: Testing images and predicted and groundtruth masks

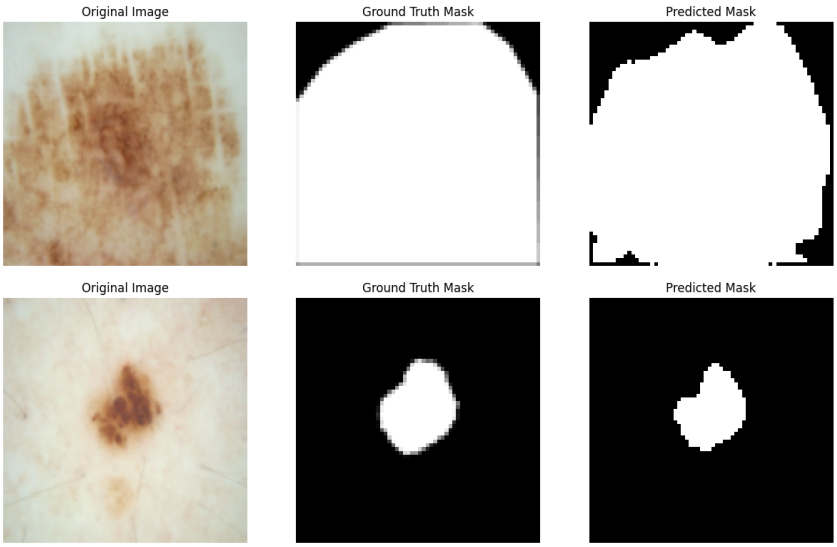


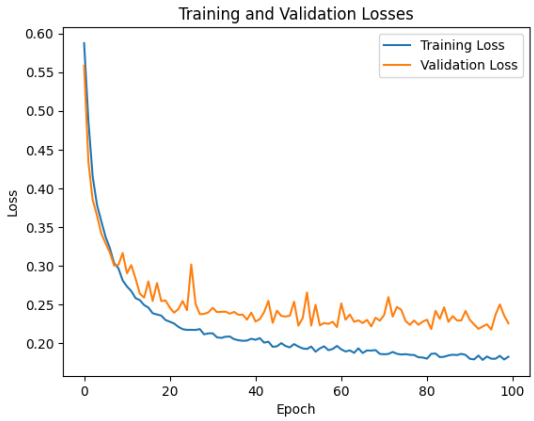
Figure 7: Testing images and predicted and groundtruth masks

With these predicted masks we can see that the model has learn the semantic segmentation well. In the first image it is able to capture the intricate boundary of the msak. In second row the predicted mask is almost similar to the groundtruth mask. In third image there is some false negative region. In some sorners the pixels are black. This is happening because difficulty in extracing the features. There may be these reasons behind the false positive and false negative results:-

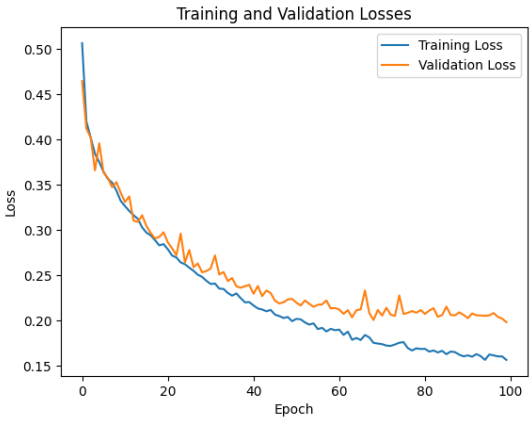
- The segmentation model may not be robust enough or may not be trained with sufficient data diversity to accurately segment all types of images.
- Images with complex backgrounds may confuse the segmentation model. This leads to false positives or false negatives.
- Objects with complex boundaries with varying colors can be challenging for the model to segment accurately.
- Inconsistent alignment or registration between the input images and their corresponding masks can result in false segmentation results.

10 Training architecture SegNet attention with atrous with data augmentation

In this section I'll elaborate the results of image segmentation on augmented dataset. Apply random crop, horizontal flip, rotate, color jitter and Gaussian noise transform as an augmentation. I kept original training images as it is and created new images with these transformations. Then combine original training examples with augmented exampled to create new big dataset.



(a) Training Loss for feature extraction



(b) training Loss for fine tuning

Figure 8: Training and Validation Loss

Both training and validation losses are decreasing this shows that the model is continuously learning. Loss is converging towards the end of 100 th epoch.

10.1 IoU and Dice score on test data

Architecture	mean IoU	Dice Score
SegNet + attetion + Atrous conv (Pre trained)	0.798	0.879
SegNet + attetion + Atrous conv (fine Tuning)	**0.81**	**0.89**

Table 2: IoU and Dice score for different architectures

In the augmented dataset I increased the dataset size by twice amount. So there are 900 original training images along with 900 augmented images for training, means total 1800 images for model training.

The mean IoU and Dice score is very good on test dataset. This is result of data augmentation. Data augmentation makes the model more robust and predictions becomes more stronger and correct. The edges are also well predicted in test images. Localization of segmentation map is also good and false results are less.

Output of testing data is shown below:

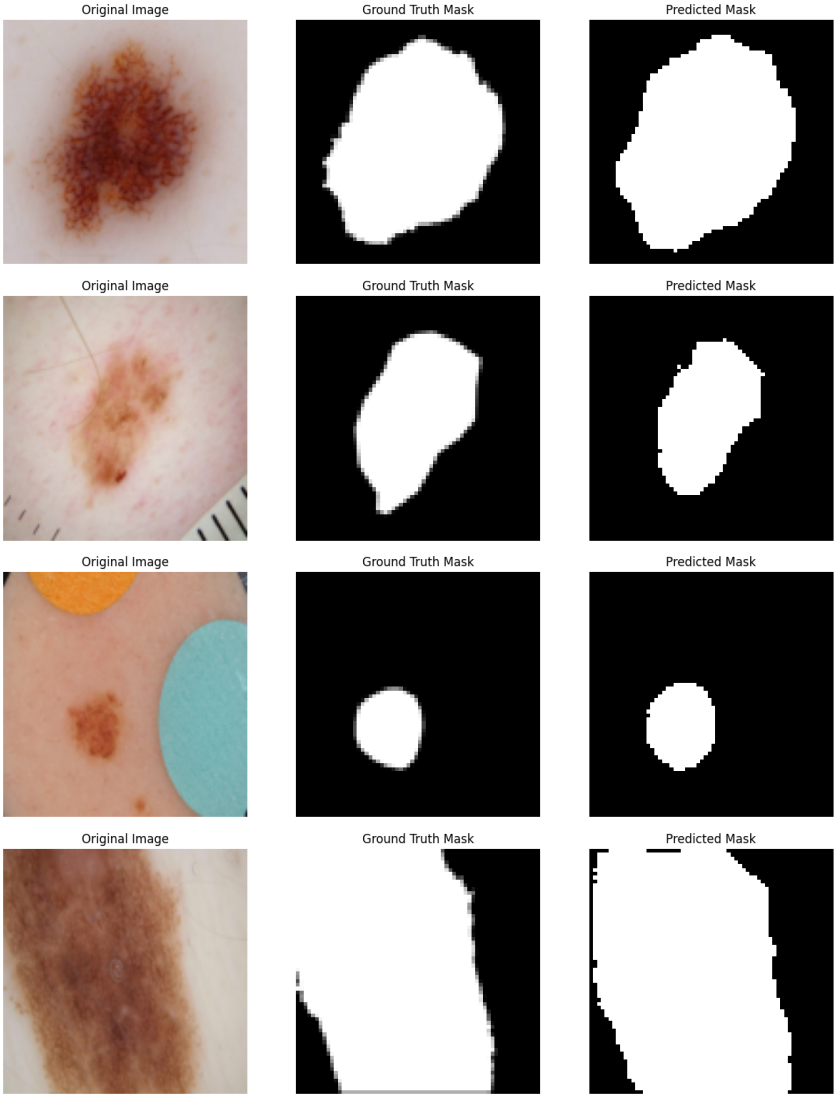


Figure 9: Testing images and predicted masks on big augmented dataset

11 Fine Tuning the encoder

In the task two it is asked to do the fine tuning of the encoder for getting the better results.

11.1 Fine tuning

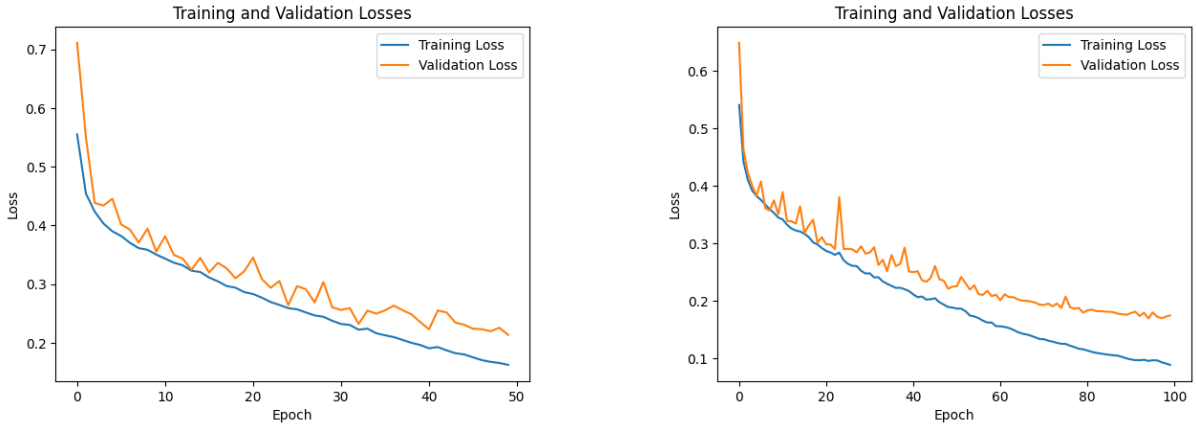
In the model fine tuning I loaded the pre-trained weights. Train the encoder with small learning rate of 0.000001 and on top of it train the docoder with learning rate of 0.0001 with BCE loss. Train the fine tuned model for 100 epochs.

11.1.1 Hyper parameter tuning

I have done the hyper parameter tuning for the fine tuned model. Initially for selecting the learning rate I varied the the value of learning rate from 0.0001 to 0.000001 for encoder. Out of these values 0.000001 was giving slightly better results. This is happening because

weights of the encoder will not change so much and will be fine tuned over pretrained weights.

Initially I trained the model for 20 epochs for checking the results of learning rate tuning. Once the learning rate is fixed I trained the model for 50 epochs to check the convergence of results. But from the loss plot it can be observed that we can still decrease the loss as it is not saturating after 50 epochs also. So finally I trained the model for 100 epochs and the results are as follows:-



(a) Training Loss for 50 epochs

(b) training Loss for 100 epochs

Figure 10: Training and Validation Loss

from 10 it can be seen that the loss is converged and model has learnt well. Training and validation loss is continuously decreasing this shows model is continuously learning new features from the image.

11.2 IoU and Dice score for fine tuned model

Mean IoU after encoder fine tuning is around 0.81 and Mean dice score is 0.88 on test dataset. From these values we can say that the model has learnt to segment the disease well. In some images again false areas are coming, the reasons for this is already mentioned in this report.

There is very good improvement in the model predictions after fine tuning. Fine tuning updates the pretrained weights with small amount so that the target task can be performed well.

Architecture	mean IoU	Dice Score
UNet	0.792	0.858
SegNet + attetion	0.70	0.79
SegNet + attetion + Atrous conv	**0.81**	**0.88**

Table 3: IoU and Dice score for different architectures after fine tuning

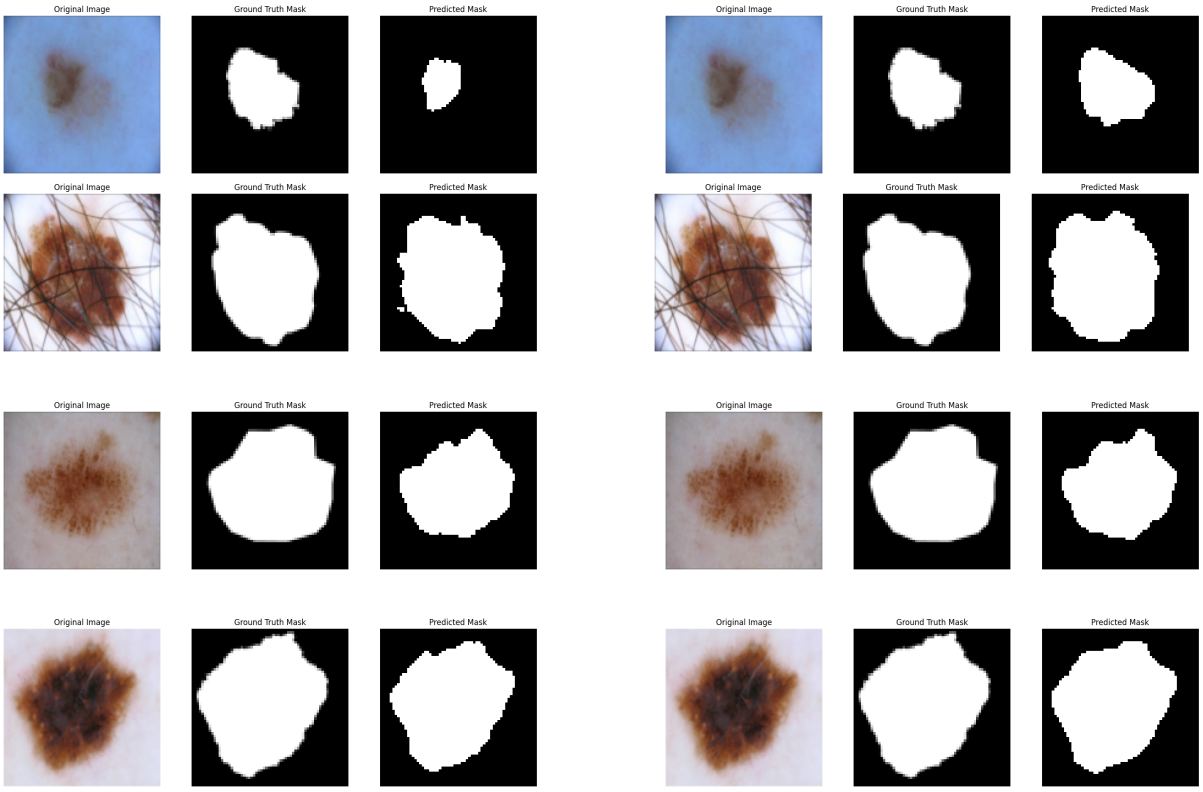
12 Comparison of Pretrained encoder vs Fine Tuning

In the task it is asked to first train the decoder with pretrained encoder. Then fine tune the encoder and train the docoder. Table below shows the detailed comparison between the pretrained model and fine tuned model.

Architecture	mean IoU	Dice Score
UNet (Pre trained)	0.77	0.858
UNet (Fine Tuned)	0.792	0.858
SegNet + attetion (Pre trained)	0.58	0.70
SegNet + attetion (Fine Tuned)	0.70	0.79
SegNet + attetion + Atrous conv (Pre trained)	0.78	0.871
SegNet + attetion + Atrous conv (Fine Tuned)	**0.81**	**0.88**

Table 4: Comparison between pretrained encoder vs fine tuning

12.1 Predicted masks for pretrained model vs fine tuned model



(a) Pretrainde encoder (b) Fine Tuning

Figure 11: Comparison Pretrained encoder vs Fine Tuning for SegNet + attetion + Atrous conv Architecture

From the image 1 in the row of predicted mask on pretrained model and fine tune model predicted mask of corresponding image it can be seen that the fine tuned model has

predicted the mask very well. Predicted mask was far away from ground truth, but after fine tuning it has captured the disese well and predicted correct mask.

From image 2 in the row pretrained model have predicted the mask well but there are some false positive in left bottom of the image. But fine tuning have removed this false positive.

From image 3 in the row it can be seen that the conour taken by fine tuned predicted mask is more close to the groundtruth mask. And in the last image both models have performed well and predicted good result.

Fine tuning has worked well because of fine adjustment of weights of network for the task of segmentation. **Mobilenet is trained on Imagenet for classification, but this model is very good at extracting the features from the image. Therefore after designing the decoder on top of mobilenet encoder model worked well for segmentation task also.**

13 References

- MobileNetV2: Inverted Residuals and Linear Bottlenecks
- Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation
- SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation
- Atrous convolution
- Loss functions in SegNet
- ChatGPT