JunHyuk Kim 2023 Report-ISIC 2017 improved Unet segmentation task.

1. Requriments

1.1. Download Dependencies

Modules that are required to run the code: tqdm, torchvision, albumentations, torch, matplotlib, sklearn, glob, PIL, os, pandas, numpy python 3.11.5 window 11

1.2. Download ISIC 2017 datasets.

The dataset is under https://challenge.isic-archive.com/data/#2017. Training data, test data, valid data is required and train ground truth, test ground truth and valid ground truth is also required for the segmenation task. The dataset should be positioned in the same folder as the python codes under data, train data images under train_images folder, and train mask data under train_masks folder. Test data should be in the test_images and test_masks and validation should be in val_images and val_masks folder.

2. Usage

Run the Train.py with python3 this will create the model.pth file. This will create predictions for each epochs and will also create the loss graph. Run predict.py to get the segmentation and the dice coefficient. This file will save all the segmentation predictions under evaluation_folder.

3. Description

3.1. Brief description of the task

Lesion segmentation is crucial in medical imaging. The code solution that is created solves the problem of the segmentation of lesion by automating it, which can benefit the medical prefessionals. The ISIC 2017 improved Unet segmentation task is implemented. The improve Unet is used and the format follows https://arxiv.org/pdf/1802.10508v1.pdf. By doing this, the accuracy and speed of segmenation can be increased significantly.

3.2. Reproducibility

After 10 epochs, the model relably provided dice score arount 80 percent.

3.3. provide example inputs, outputs and plots of your algorithm

For every epoch, the validation set is used to test the segmenation.

3.4. Describe any specific pre-processing you have used with references if any. Justify your training,

3.4.1. pre-processing I used transformation to increase the number of samples the model is trained on. For transformation the albumentations module was used. This makes the transformation much eaiser. Transformation used are, rotate, horizontalflip, veritcalflip. Also the values were normalized. I also resized the images into 256x256 sizes to increase the performance of training and also because my model works on that sizes.

3.4.2. Training The training is done in batch size of 16 with 10 epoches. The opitmizer used is ADAM and the loss function used is Dice loss function. The schaduler was used to reduce the learning rate at the end fo the function. The training accuracy and saved predictions are from validation imagesets. GradScaler is used to make the training faster.

3.5 Commentation Each class and functions/methods are commented. Comment includes a brief description of the function of the class,method or the function and followed by the parameters it recieves and the return values it returns.

4. Algorithm/ImprovedUnet

This program uses improved Unet to segment the differnt types of skin conditions such as melanoma. The improved Unet strucuture explained

4.1. Context layer and Localization Layer.

Context Layer consists of 2d convolutional layer with kernel size 3, InstanceNorm2d, LeakyReLU and dropout layer with p value of 3. After that, another 2d convolutional layer with kernel size 3 is applied followed by instanceNorm2d and LeakyReLU. Context layer normally recieves the same output as input channel, which means goingthough context layer doestn really change the dimension of the tensor. Localization Layer consists of 2d convolutional layer with kernel size 3, InstanceNorm2d, LeakyReLU. After that, another 2d convolutional layer with kernel size 1 is applied followed by instanceNorm2d and LeakyReLU. Localization layer changes the dimension of the tensor given by halving it. This is because it recieves the concatination of skip_connection and the previous output from the previous layer.

4.2. Layer 1 to Layer 5.

Firstly,2d convolutional layer is applied to the input tensor X, and the output is saved. First convolutional layer changes the dimension of the tensor to 3 to 64, but all the other pathways doubles the dimension of the tensor. The output if fed into the the context layer, and the output of the context layer and the first convolutional layer is added element wise using the “torch.add()” function. The output is saved in the name of skip_connection_n(where n is the layer which the context pathway is in). Than this countinues 4 more times except for the 5 th time the skip connection is not saved.

4.3. Layer 5 to Layer 1.

In the layer 5 after applying the context pathway for 5 times, localization pathway is applied. Localization pathway consists of first upsampling the tensor(which doubles the size of the image) and a convolutional layer with kernal size of 1, which halves the amount of features the tensor has. After this, the skip layer which corresponds to the layer which the localization layer is positioned is concatinated to the previous output. Than the output is fed into the localization layer, which will halve the amount of features the tensor has. This is because the number of feature is doubled after concatination. This continues until the the ouput reaches the layer 1, where instead of localization a normal convolutional layer with kernel size 3 is applied. This doesnt change the amount of the feature map. During the localization pathway, in Kayer 3 localization and Layer 4 localization, and after the final convolutiol layer, segmentaion layer which is just a convolutional layer with out channel of 1 is appied and the output is saved under the name of segmentation 1, segmentaion 2 and segmentation 3. Segmentation 1 is upscaled and added element wise to segmentaion 2, and the combination of those are upscaled and added to segmenation 3 element wise. Finally the combination of all are ran though Sigmoid function and the model returns the output.

5 Appendix.

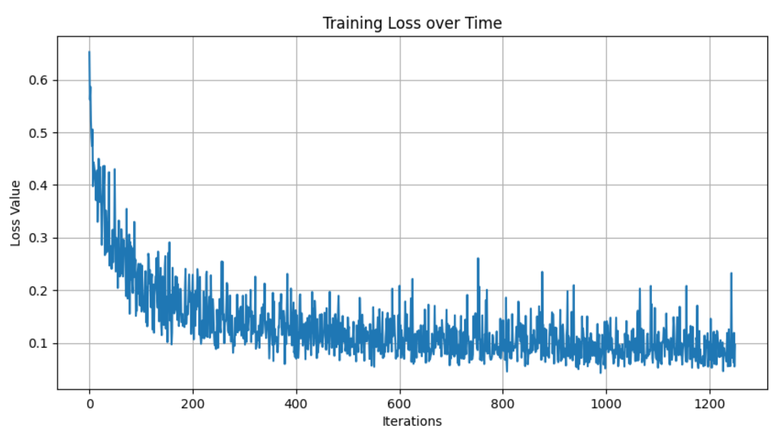


Figure 1: loss plot

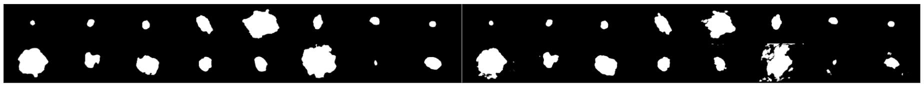
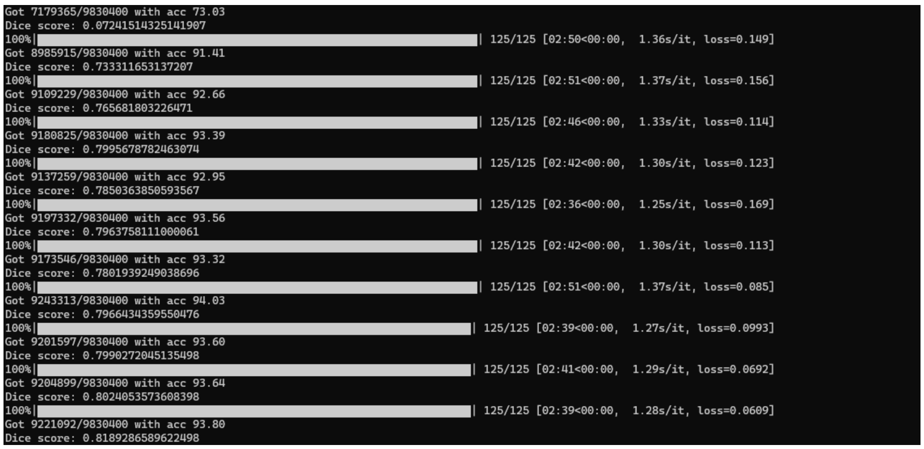


Figure 2: train_valid_out



```
Got 7179365/9830400 with acc 73.03
Dice score: 0.07241514325141907
100%| | 125/125 [02:50<00:00,  1.36s/it, loss=0.149]
Got 8985915/9830400 with acc 91.41
Dice score: 0.733311653137207
100%| | 125/125 [02:51<00:00,  1.37s/it, loss=0.156]
Got 9109229/9830400 with acc 92.66
Dice score: 0.765681803226471
100%| | 125/125 [02:46<00:00,  1.33s/it, loss=0.114]
Got 9180825/9830400 with acc 93.39
Dice score: 0.7995678782463074
100%| | 125/125 [02:42<00:00,  1.30s/it, loss=0.123]
Got 9137259/9830400 with acc 92.95
Dice score: 0.7850363850593567
100%| | 125/125 [02:36<00:00,  1.25s/it, loss=0.169]
Got 9197332/9830400 with acc 93.56
Dice score: 0.7963758111000061
100%| | 125/125 [02:42<00:00,  1.30s/it, loss=0.113]
Got 9173546/9830400 with acc 93.32
Dice score: 0.7801939249038696
100%| | 125/125 [02:51<00:00,  1.37s/it, loss=0.085]
Got 9243313/9830400 with acc 94.03
Dice score: 0.7966434359550476
100%| | 125/125 [02:39<00:00,  1.27s/it, loss=0.0993]
Got 9201597/9830400 with acc 93.60
Dice score: 0.7990272045135498
100%| | 125/125 [02:41<00:00,  1.29s/it, loss=0.0692]
Got 9204899/9830400 with acc 93.64
Dice score: 0.8024053573608398
100%| | 125/125 [02:39<00:00,  1.28s/it, loss=0.0609]
Got 9221092/9830400 with acc 93.80
Dice score: 0.8189286589622498
```

Figure 3: train_cmd

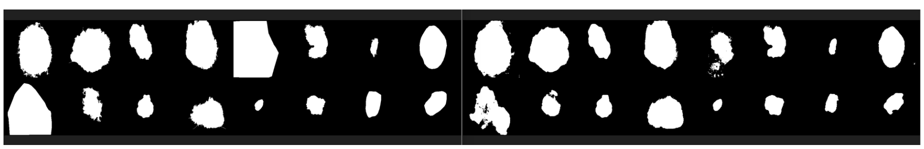
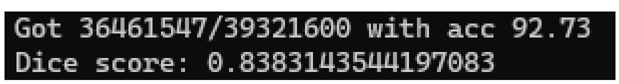


Figure 4: prediction_images



```
Got 36461547/39321600 with acc 92.73
Dice score: 0.8383143544197083
```

Figure 5: predict_cmd_result

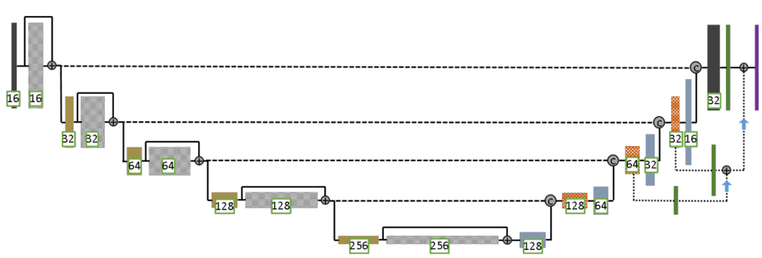


Figure 6: My Image Alt Text