Homework 3: Medical Image Segmentation

Step 1: Data

1. There are 160 training samples (80 images and 80 masks of those corresponding images) and 40 testing samples (20 images and 20 masks of those corresponding images), so overall a 200.

2. The goal is to predict the segmentation of Retina Blood Vessels.

3. All the images in the dataset have dimensions, 512\*512 or more.

4. No, the dataset doesn’t have any missing information.

5. The labels in this dataset are binary masks where blood vessel pixels are marked as 1, and background pixels are labelled as 0.

6. 70% training, 15% validation and 15% testing

7. Data pre-processing steps include resizing images to 256\*256, reading and joining the images and masks, normalization, and Data Augmentation.

Step 2: Model

|  |  |  |
| --- | --- | --- |
| Model | IoU | Dice |
| U-Net 2 layers | 0.68910 | 0.81571 |
| U-Net 3 layers | 0.66709 | 0.79948 |
| U-Net 4 layers | 0.68669 | 0.81393 |

Step 3: Objective

To train UNet models with normalization, I’ve used ‘binary crossentropy’ and for models without normalization, I have defined a new type of loss function called Tversky\_loss which is a custom loss function used in binary image segmentation tasks. It quantifies the dissimilarity between the predicted segmentation (y\_pred) and the ground truth (y\_true) by considering both false positives and false negatives, with adjustable weights (alpha and beta) to emphasize different types of errors. Changing the loss function was necessary as binary\_crossentropy was not giving the desired scores for images not normalized.

Step 4: Optimization

I will be choosing Adam for optimization process. This is because based on empirical observations and heuristics, this is the best to train a U-Net model like the retinal blood vessel segmentation dataset. Adam has various features like adaptive learning rate, bias correction, and momentum to make it a top choice for training of U-Net models.

Step 5: Model Selection

|  |  |  |
| --- | --- | --- |
| Model | Dice, IoU with normalization | Dice, IoU without normalization |
| U-Net 2 layers | Dice: 0.816, IoU: 0.689 | Dice: 0.7194, IoU: 0.563 |
| U-Net 3 layers | Dice: 0.799, IoU: 0.667 | Dice: 0.661, IoU: 0.557 |
| U-Net 4 layers | Dice: 0.814, IoU: 0.687 | Dice: 0.335, IoU: 0.202 |

To avoid overfitting or underfitting, I have added several things like:

* Data Augmentation
* Weight Regularization (Use of L2)
* Adding dropout layers
* Resizing all images to a constant dimension.

Step 6: Model Performance

The Dice and IoU plots, and input images screenshots are there in the Screenshots folder submitted with this word file, with the U-Net model weights as the name of each screenshot.

*My highest performed U-Net Model was for the U-Net with 2 layers with normalization, with a dice of 0.816 and IoU of 0.689.*