REPORT FOR ASSIGNMENT 3 – INTRO TO DEEP LEARNING – Doan The Vinh 20210940

1. Transformations used include

No	Transformation	Purpose
1	Resize	To help fit the image batch into the model. Mask is transformed correspondingly.
2	Random Crop	To help further fit the image batch into the model, while also helping to improve training speed. Undesirable side effects include losing information on the neopolyp and non-neopolyp segmentation of the images. Mask is transformed correspondingly.
3	Random Horizontal Flipping	Flip image horizontally. Does not affect semantic meaning of the image. Mask is transformed correspondingly.
4	Random Vertical Flipping	Flip image vertically. Does not affect semantic meaning of the image. Mask is transformed correspondingly.
5	Random Rotation	Rotate the image by a small degree. Does not affect semantic meaning of the image. Mask is transformed correspondingly.
6	Random adjusting of brightness, contrast, and saturation	To diversify the dataset. Effects are minor, so does not affect semantic meaning of the image.
7	Gaussian Blur	To blur the images with varying kernel sizes. Effects are appropriate, so does not affect semantic meaning of the image.
8	Random Perspective	To view the image warped in a different perspective. Does not affect semantic meaning of the image
9	Normalize	Normalize pixel values
10	To Tensor	Turn image to tensor

- These transformations are applied in full to the training set (80% of the entire set), but for the validation set, only Resize, Random Crop, Normalize, and To Tensor are used.
- The transformed training set are concatenated with the original training set to create a training set of 1600 instances.

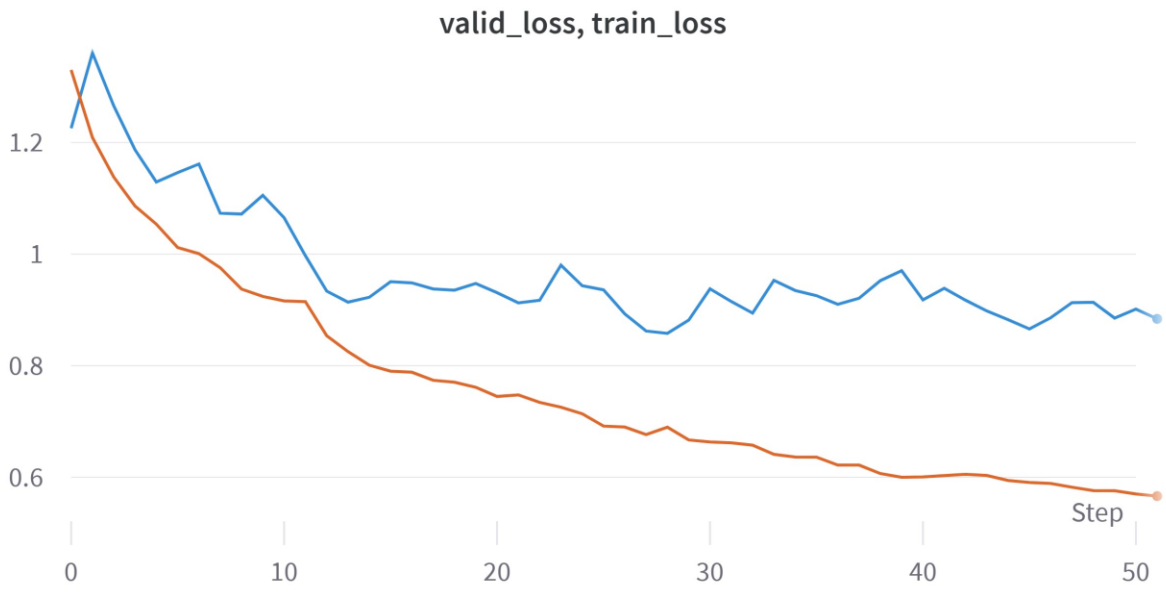
2. Changes in the model architecture include
   - The backbone (encoder) of the architecture is an implementation of the **ResNet-34** model. The encoder is divided into 4 layers, each of which has its own standard blocks of Conv2d (kernel_size=3) – BatchNorm2d – ReLU.
     - The number of blocks per layer: 3-4-6-3
     - The number of channels as output per layer: 64-128-256-512
     - BatchNorm2d used has parameters: eps=1e-05, momentum=0.1

- The decoder consists of multiple blocks. At each layer, the output of the encoder is concatenated with the result of the previous layer in the decoder. This gets passed through an **Attention module**, where it goes through **channel squeeze** and **spatial squeeze** (to enhance extracting of features). There are four such blocks, before a final Conv2d layer takes the number of channels back to 3 (num_classes)

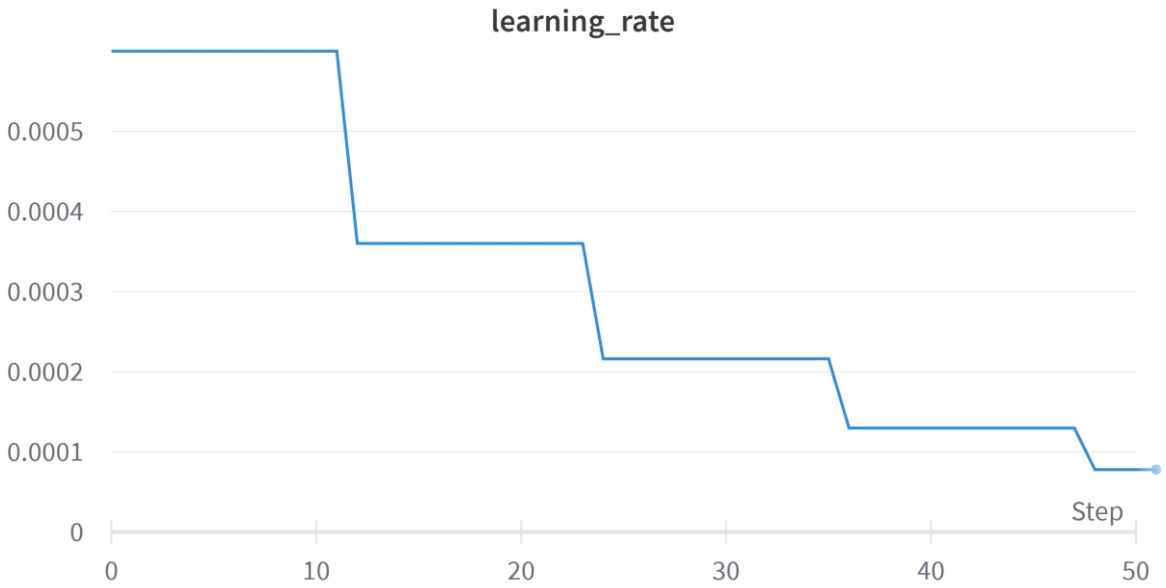
- Training-wise, I used a StepLR scheduler, with an initial learning rate of 6e-04, step_size=12, gamma=0.6. Number of epochs is set to 200, but I stopped training around epoch 50 due to potential signs of overfitting (refer to the below plot). I also changed the weights to [0.5, 0.45, 0.05], because I've seen from previous attempts that my models would often over-predict non-neopolyp (class 1).

- Beside architecture, modifications are made to the code to render it more object-oriented, thus enhancing modularization and maintainability.
- In order to deal with the problem of "CUDA out of memory", I repeatedly cleared cache and delete used tensors to free memory. I also put the model in a DataParallel module to increase training speed (though not for inference)

3. Plots of training and validation losses



Orange: training loss
Blue: validation loss



4. Github repository & Checkpoint:

   vincent-doan/unet-for-neopolyp-segmentation (github.com)

   checkpoint