Simon Fraser University

Image segmentation

Johannes Merz, Ali Mahdavi-Amiri, Aryan Mikaeili
January 21, 2022

1 U-Net: Segmentation using deep learning

In the last years, a number of neural networks for image segmentation were designed with considerable success. One of the most prominent attempts was the U-Net by Ronneberger et al. (2015) [1]. The name of the network stems from the symmetric shape of the architecture. As can be seen in Figure 1, a contracting path is followed by a symmetric expansive path.

In this assignment you should implement the U-net architecture for cell image data segmentation using PyTorch. You have been provided with a framework that helps you define your model and train/test it. In the following sections the components of the architecture will be described.

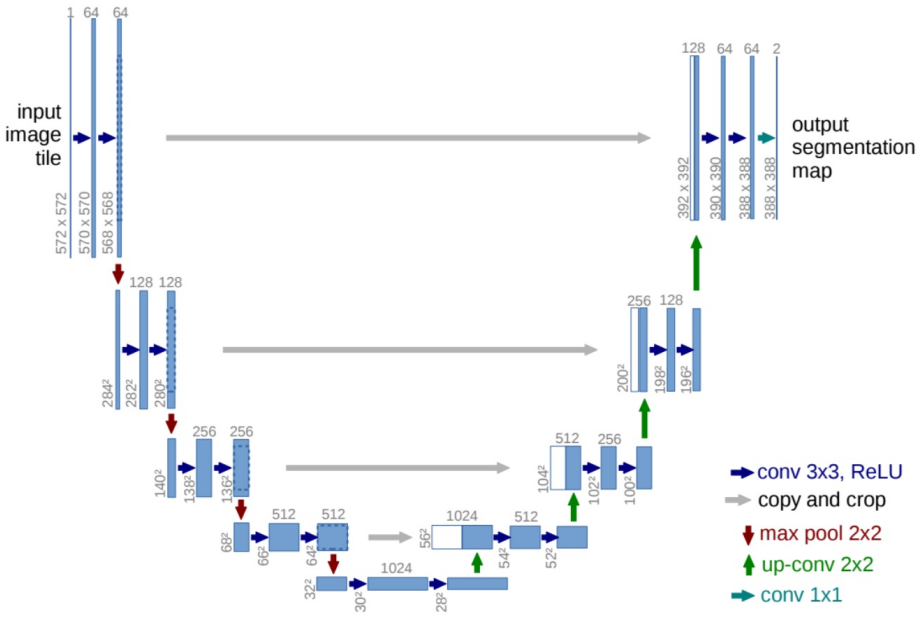

Figure 1: U-Net architecture

1.1 The Convolutional blocks (3 points)

Both the contracting path and the expansive path contain multiple convolutional blocks in which the input feature map goes through the following:

1. a 3×3 un-padded convolution layer which takes a feature map with $input\_channel$ number of channels and outputs a feature map with $output\_channel$ number of channels
2. a ReLU activation function
3. another 3×3 un-padded convolution layer which keeps the number of channels unchanged
4. a Batch normalization layer
5. a ReLU activation function

In the framework complete the class $twoConvBlock$, which implements these convolution blocks. You will later use these blocks to implement the contracting and expansive paths.

1.2 The contracting path (5 points)

This part of the architecture includes everything until the first green arrow in figure 1. As shown in the figure in each step the input feature map goes through a convolution block and then a 2×2 max-pooling layer sequentially. In each block the number of channels doubles [1].

In order to implement the contracting path, complete the $downStep$ class in the framework.

1.3 The expansive path (5 points)

The output of the contracting path, is a feature map which described plainly, summarizes the content of the input image. Now given this feature map, we want to upsample it to the original image size to localize the segmentation map.

As shown in figure 1, in each step the input feature map goes through a upsampling layer (implemented with transpose convolutions) and a convolutional block which you implemented earlier. Notice that in each upsampling convolution step, the output channel will be half the input channel, and the input of each convolution block comes from concatenating this feature map with a skip connection feature map which comes from the corresponding contracting path step (the gray arrows in figure 1).

In order to implement the expansive path, complete the $upStep$ class in the framework.

1.4 The dataset (5 points)

Now that we have the model of the network, we can prepare the data that we want to use for training. The framework already provides you with a dataset. The dataset includes a number of images and corresponding label masks (targets). It is your task to load them by completing the code in the $Cell\_data$ class in the framework. When loading the dataset there are some things that need to be considered. We are using grayscale images, so there is only one color channel. Also, before using the images, they need to be normalized to $[0, 1]$. This improves the learning performance and the stability of our network.

1.5 Data augmentation (5 points)

As you can see the size of the data is too small for training a neural network with a huge number of parameters. That is why that in many cases that we do not have enough data we augment our dataset by applying different transformations to the available dataset. In your code try at least three of the following techniques:

1. Horizontal/Vertical flip
2. Zooming
3. Rotation
4. Apply gamma correction
5. the authors of UNet heavily relied on non-rigid transformations for data augmentation. In their paper they describe their technique like this: We generate smooth deformations using random displacement vectors on a coarse 3 by 3 grid. The displacements are sampled from a Gaussian distribution with 10 pixels standard deviation.

[1] Except for the first block which takes 1 channel and outputs 64 channels

1.6 Experiments and report (4 points)

Train your model and the data and run experiments with it. Then write a report containing the following information

- Report the resulting segmentation of the test images
- What was the size of the images you used for training? you can rescale your images for faster training and better memory usage. For the final results, use at least 128×128 images.
- Which data augmentation strategies did you use?
- Did you deviate from the original architecture? If so mention the changes.
- how many epochs did you train? provide plots for the train-test losses.
- what batch size and learning rate did you use?
- How long did the training take?

References

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.