Practical 2- Machine Learning in Medecin

Cao Hieu Vinh

March 2025

Practical 2

1 Deep Learning for this task

1.1 Description of the U-Net

The U-Net is a convolutional neural network (CNN) architecture commonly used for image segmentation tasks. The U-Net architecture is characterized by its U-shaped design, which consists of an encoder path and a decoder path. The encoder path captures the context and spatial information from the input image, while the decoder path performs upsampling and reconstructs the segmented image.

Here's a detailed description of the U-Net architecture:

1. **Encoder Path:**
   - The input to the U-Net is an image, typically of variable size.
   - The encoder path starts with a series of convolutional layers followed by rectified linear unit (ReLU) activations, which extract features from the input image.
   - These convolutional layers are typically configured to have a small filter size (e.g., 3×3) and are followed by max pooling layers with stride 2, which down-sample the feature maps and increase their receptive field.
2. **Decoder Path:**
   - The decoder path begins with up-sampling operations, such as transposed convolutions or bilinear up-sampling, which increase the spatial resolution of the feature maps.
   - **Concatenation:** At each up-sampling step, the feature maps from the corresponding encoder path level are concatenated with the up-sampled feature maps.
   - The concatenated feature maps are then passed through a series of convolutional layers with ReLU activations.
3. **Output:**
   - The final layer of the U-Net is a 1×1 convolutional layer followed by a sigmoid or softmax activation function, depending on the number of classes for segmentation.
   - This layer produces the segmented image or probability map, where each pixel represents the probability of belonging to a particular class or being part of the segmented object.

1.2 Preprocessing Methods

As preprocessing, I first resized the images to 128×128 for the implemented CNN. Also, after resizing the images, I normalized them. As we know, normalizing the pixel values of the images can help in reducing the effects of lighting variations and improve the convergence of the network during training. Common normalization techniques include subtracting the mean and dividing by the standard deviation of the image dataset or scaling the pixel values to a specific range, such as $[0, 1]$. I scaled the images into $[0, 1]$ by dividing the pixels by 255.

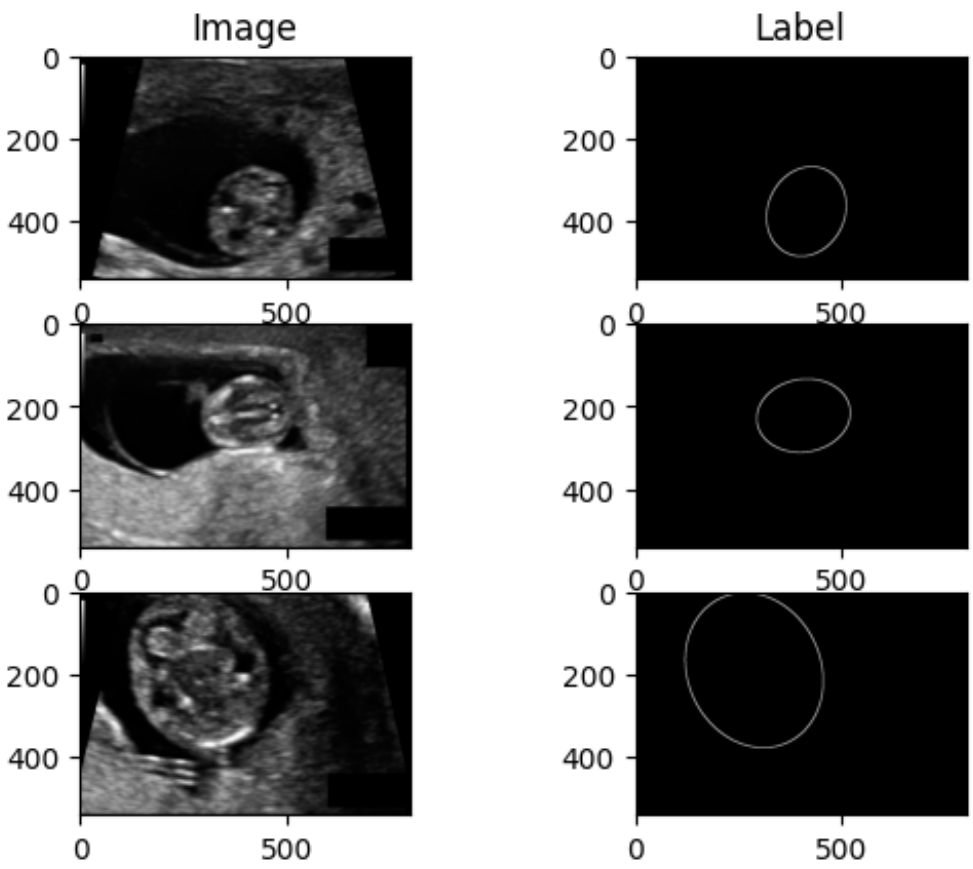

Figure 1: Sample of the dataset

1.3 Code implementation

For training the model, I used 80% of the dataset as the training set and 20% for the test set. I set the batch size to 16 as mentioned in the article and also used the Adam optimizer for training. The loss function I used is binary cross-entropy. This model was trained for 25 epochs,the accuracy of the model is about 88.97% for the test set and 88.7% for the validation set.

The result of the model for a random datapoint is as follows:

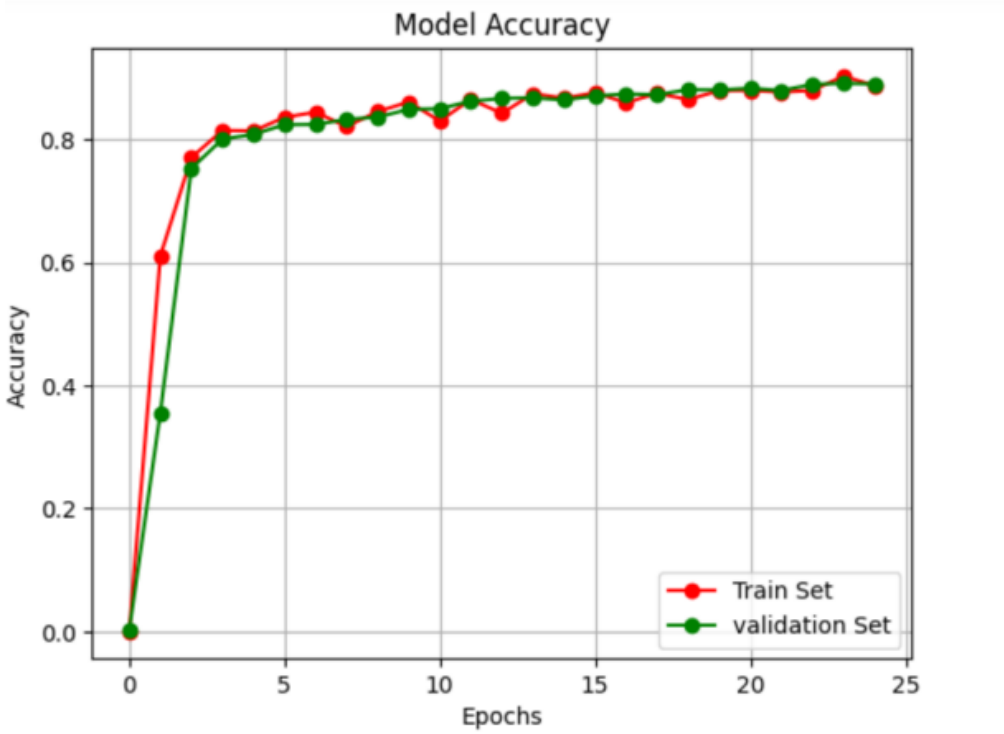

Figure 2: Accuracy of U-Net Model

The result of the model for a random datapoint is as follows:

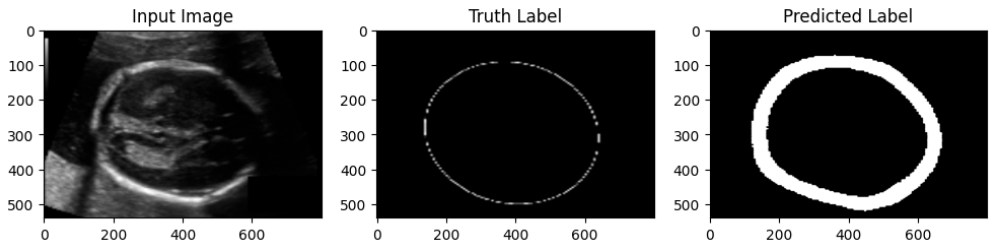

Figure 3: Model prediction with a sample from test set