**Image Segmentation**

Amit Lakshmikant Chaudhary1,

Ankit Anand2

*Department of Computer Science, University of Alabama at Birmingham*

*1720 University Blvd, Birmingham, AL, United States 35294*

1[achaudha@uab.edu](mailto:achaudha@uab.edu)

2<ankitk@uab.edu>

Contents

Introduction ……………........................................................................................................................... 3

U-Net Architecture ……........................................................................................................................ 3

Double U-Net Architecture ................................................................................................................... 4 Data .................................................................................................................................................... 5

Requirements for the Project ……………................................................................................................ 5

Data Exploration.................................................................................................................................... 5

Procedure using U Net and double U Net ............................................................................................. 6 Error function ………................................................................................................................................ 9 Training on Stem Cells in a Petri dish.................................................................................................... 10 Results …............................................................................................................................................. 10 Challenges Faced .............................................................................................................................. 11

References ………………………………………………………………………………………………………………………………………. 11

Introduction

We have a very huge image dataset that needs to be examined not only in the medical sector but also in a real-time scenario. A crucial stage in drug testing and discovery is determining the total number of colonies of induced cells in a petridish. Counting colonies manually is tiresome, therefore we need to automate it with image segmentation. Our model explains how deep learning models for image segmentation can be used to identify cell colonies.

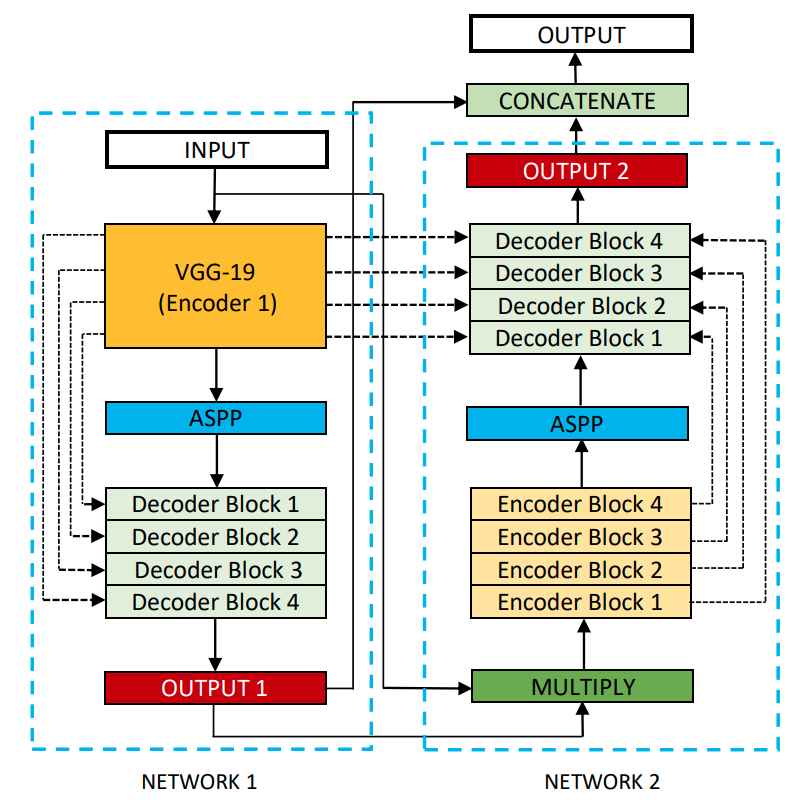
Although a general convolutional neural network focuses on image classification, using an image as input and a single label as output, in biological applications, we must not only determine whether there is a sickness but also locate the anomaly. This is a problem that we can solve with UNet. It can localize and discern borders since it does classification on each pixel.

Simple U-Net Architecture

Chart, box and whisker chart

Description automatically generated

Double U-Net Architecture



The encoder and decoder were the two fundamental components of the introduced design. The covenant layers are the most important part of the encoder, followed by the pooling procedure. It is used to extract the image's factors. To allow for localization, the second part decoder employs transposed convolution.

The architecture is symmetric and consists of two major parts — the left part is called contracting path, which is constituted by the general convolutional process, and the right part is expansive path, which is constituted by transposed 2d convolutional layers.

Data

The file contains two folders among which first contains all the images of stem cells in a petri dish while on the other hand the second file contains all the segmentation masks of all the images(First folder).

Data

* Images
* Segmentation\_mask

Requirements for the Project

Formal understanding of deep learning and machine learning methodologies using Python, TensorFlow, Keras, loss functions and optimizers.

Data Exploration

With the use of graphs, histograms and seaborn library we looked at different distributions of our data –

A picture containing text

Description automatically generated Graphical user interface, chart

Description automatically generated

Step by Step Detection

To begin with, we'll require to import all the required libraries. We have a function img\_read to read all the image files from our directory and assigning all the images with their corresponding masks using a img\_meta\_obj

Sample image and their mask -

A picture containing cup, indoor, Petri dish, tableware

Description automatically generated A picture containing outdoor object

Description automatically generated

A picture containing cup, indoor

Description automatically generated A black and white image of a tree

Description automatically generated with low confidence

Sam

Example of segmentation mask on top of image –

A picture containing indoor, dishware, tableware

Description automatically generated

Our data and mask after preprocessing will look like -

A picture containing indoor, cup, food, milk

Description automatically generated

Since, we are working with images we need to pre-process all the images to similar size so that we can feed it into the model. We created our own dataloader to resize all the images and then normalizing it.

To train our model, we will be using UNet model. UNet does not have fully connected layers. The segmentation map contains only the pixels for which the full context is available in the input image. An overlap-tile strategy is used to segment arbitrarily images and the missing input data is extrapolated by using a padding of the input image.

Our encoder consists of repeated 2 3x3 convolutions (unpadded), followed by a ReLU and a 2x2 max pooling with stride 2 for downsampling. In the next group of layer (encoder), the number of feature channels is doubled.

The decoder path is a series of upsampling operations of the feature map followed by a 2x2 convolution that halves the number of features channels. Each step is concatenated with the corresponding feature map from the decoder. The feature maps are actually cropped since the border is lost because of the convolutions. Finally the groups of layers (step) are followed by two 3x3 convolutions and a ReLU.

The difference between Double U-Net and U-Net is in the second network that we are using ASPP and squeeze-and-excite block.

In the NETWORK 1, the input image is fed to the modified U-Net, which generates a predicted mask (Output1). We then multiply the input image and the produced mask (Output1), which acts as an input for the second modified U-Net that produces another mask (Output2). Finally, we concatenate both the masks (Output1 and Output2) to see the qualitative difference between the intermediate mask (Output1) and final predicted mask (Output2).

The first encoder in DoubleU-Net (encoder1) uses pretrained VGG-19, whereas the second encoder (encoder2), is built from scratch. Each encoder tries to encode the information contained in the input image. Each encoder block in the encoder2 performs two 3 × 3 convolution operation, each followed by a batch normalization. The batch normalization reduces the internal co-variant shift and also regularizes the model. A Rectified Linear Unit (ReLU) activation function is applied, which introduces non-linearity into the model. This is followed by a squeeze-and- excitation block, which enhances the quality of the feature maps. After that, max-pooling is performed with a 2 × 2 window and stride 2 to reduce the spatial dimension of the feature maps.

We use two decoders in the entire network, with small modifications on the decoder as compared with that of the original U-Net. Each block in the decoder performs a 2 × 2 bi-linear up-sampling on the input feature, which doubles the dimension of the input feature maps. Now, we concatenate the appropriate skip connections feature maps from the encoder to the output feature maps. In the first decoder, we only use skip connection from the first encoder, but in the second decoder, we use skip connection from both the encoders, which maintains the spatial resolution and enhance the quality of the output feature maps. After concatenation, we again perform two 3 × 3 convolution operation, each of which is followed by batch normalization and then by a ReLU activation function. After that, we use a squeeze and excitation block. At last, we apply a convolution layer with a sigmoid activation function, which is used to generate the mask for the corresponding modified U-Net.

Text

Description automatically generated

We created the following build\_model function to utilize the Double UNet architecture, code snippet –

Text

Description automatically generated

And created the model object with the input parameter of image size that is (256,256,3) –

model = build\_model((256, 256, 3))

ERROR FUNCTION –

We used dice\_function as our loss function, to measure overlap between images and their masks. Dice function ranges from 0 to 1, where 1 means perfect overlap. Because our target mask are binary, we effectively zero-out any pixel from our predictions; a higher value for the expression, which is in the numerator, leads to a better dice\_coefficient.

Text, letter

Description automatically generated

Training the model –

We used following hyperparameters for our model –

BATCH\_SIZE = 5

BUFFER\_SIZE = 1000

Epoch = 5

Steps/epoch = train\_size // BATCH\_SIZE

Validation step = val\_size // BATCH\_SIZE

Learning rate = 10-3

Metrics = accuracy

Optimizer = AdamOptimizer

Adam optimizer - This optimization technique adjusts learning rate according to squared gradients. It is an adaptive learning rate method, hence it computes specific learning rates for different parameters. Adam optimizer is the base for future model tuning.

RESULTS –

We ran a total of 5 epochs on CPU and with every epoch our accuracy increased and our best model was saved with an accuracy of 97.07 %. The accuracy on validation dataset was

We predicted the segmentation for a few validation images to manually see the accuracy of our best saved model.

Sample of predicted images-

A picture containing cup, indoor, orange

Description automatically generated A picture containing indoor, dishware

Description automatically generated

CHALLENGES FACED –

1. Training the data took longer hours.

2. Changing the pixel size to send it to the model. As the input data and output data size should match for the next encoder in double U-Net model, where we were getting errors.

3. We were able to run only 5 epochs as it was taking too long to run a single epoch. So, our accuracy could not reach to its best.

References –

1. <https://arxiv.org/pdf/2006.04868v2.pdf>/
2. <https://analyticsindiamag.com/my-experiment-with-unet-building-an-image-segmentation-model/>
3. <https://www.jeremyjordan.me/semantic-segmentation/>