Cell nuclei segmentation using U-Nets

Yashwant Orru, Rahul Chunduru

160050066, 160050072

June 2020

1 Introduction

Cell nuclei segmentation is commonly used in microscopy image analysis. For example, the morphological changes in nuclei is crucial in present day tumor analysis. In addition, the count of nuclei to cytoplasm ratio in a cell are often used by pathologists. Cell nuclei segmentation can be viewed as a semantic segmentation, which is a classic problem in Computer vision. Semantic segmentation in general is a more complicated problem than classification, where the latter requires a single classification label over the whole input, the former requires an inference label over each pixel for localizing the object. In practical settings, resources and time are a major constraints in a typical third world hospital setting. Therefore, a lightweight, low-on-computation solution to this task is very much desired.

In this project, we explored different deep learning segmentation techniques for Cell nuclei and implemented variants [1] of U-nets architecture from scratch. We then evaluated their performance using different metrics.

2 Problem Statement

In this project, we considered the **2018 Data Science Bowl[1]** task. On a given cell tissue microscopy medical image, the task is to determine the pixels which constitute nuclei.

3 Data set description

We used stage 1 training set of the Data Science Bowl task. It consists of 670 training images and 65 test images. Each training image is accompanied with a 0-1 masks for every nuclei it contains.

[1] Our implementation and testing code is available at `https://github.com/Rahul-chunduru/Unets-segementation`

4 Methods

Here, we shall describe commonly used deep learning methods for segmentation tasks.

4.1 Data Augmentation

Data augmentation, i.e, artificially adding images to data set by performing transformations both geometric and noise-additive on the given data set, makes the trained model robust to these transforms. We used **'rot90'** and **'add_random_brightness'** transforms which rotates input image by 90 deg. and adjusts brightness by a random factor.

4.2 Network Architecture

We considered the U-net architechture for this task.

UNET[2] The Network Architecture illustrated in Figure 1 consists of contracting and expansion path. The contraction path(encoder) which tries to capture the context of the image is a stacking of convolution and max pooling layers. The expansion path(decoder) is used to enable precise localisation using transposed convolution and skip connections to allow the flow of information from encoder to decoder. The Encoder consists of repeated application of two 5*5 convolutions followed by 2*2 max pooling operations for down-sampling. After each down-sampling step the number of convolution features get doubled. Every step of the decoder consists of up-sampling of the feature map with 2*2 convolutions(up-convolution) with the half the number of features, a concatenation of the corresponding feature map in the encoder and 3*3 two convolutions with dropout layers added after down-sampling in encoder and after concatenation in Decoder.

UNET++[3] This architecture, shown in Figure 2 is an enhanced UNET with added convolutions layers in the skip pathways to bridge the semantic gap between the encoder and decoder feature maps. Instead of direct skip connections in UNET, in UNET++ they undergo a dense convolution block whose number of convolutions depend on the pyramid level. Each convolution layer is preceded by concatenation of previous convolutions block of the same block and upsampled output of lower dense block. The hypothesis is that optimization would face a easier optimisation problem when the encoder feature maps and corresponding decoder feature maps are semantically similar.

4.3 Loss metrics

The model predicts probability of being a part of cell nuclei for every pixel in the image and trained on three different loss functions. With $\hat{\mathbf{p}}$ be the the predicted probability and $\mathbf{p}$ as true probabilities loss functions are defined as follows.

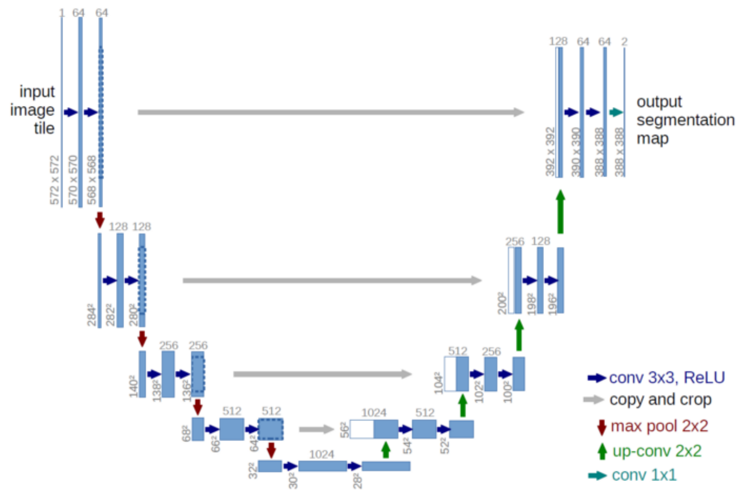

Figure 1: U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

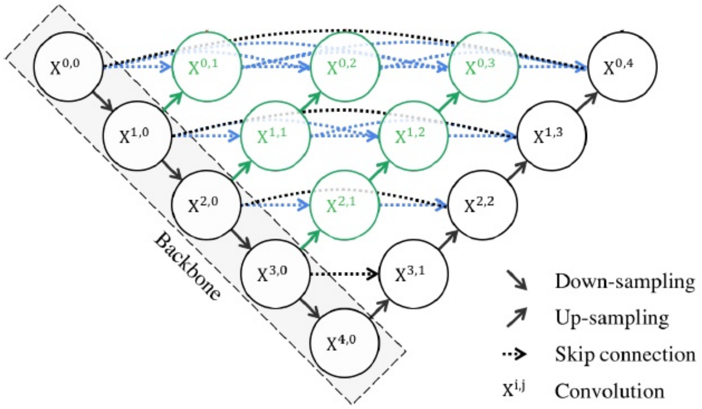

Figure 2: In the graphical abstract, black indicates the original U-Net, green and blue show dense convolution blocks on the skip pathways

Cross Entropy Loss : The loss is described as follows:

$$CE(p,\hat{p}) = -((plog(\hat{p}) + (1-p)log(1-\hat{p}))$$

Focal Loss : Focal loss tries to down-weight the contribution of easy examples so that the CNN focuses more on hard examples.

$$Fl(p, \hat{p}) = -((1-\hat{p})^{\gamma} p log(\hat{p}) + (\hat{p}^{\gamma}(1-p) log(1-\hat{p})))$$

Dice Loss : Dice Coefficient is similar to jacard index(Intersection over union, IoU) and dice loss is defined in terms of dice coffiecient.

$$DC = \frac{2TP}{2TP + FP + FN} = \frac{2|X \cap Y|}{|X| + |Y|} DL(p, \hat{p}) = 1 - (\frac{2p.\hat{p}}{p + \hat{p}})$$

4.4 Inference

Given a test input image, the model output a probability image matrix. A threshold of 0.5 to determine whether a pixel is in a nuclei or not.

5 Results

5.1 Training statistics

For each of the U-net variant and loss function, we plot the training statistics. Using **'Dice Loss' :**

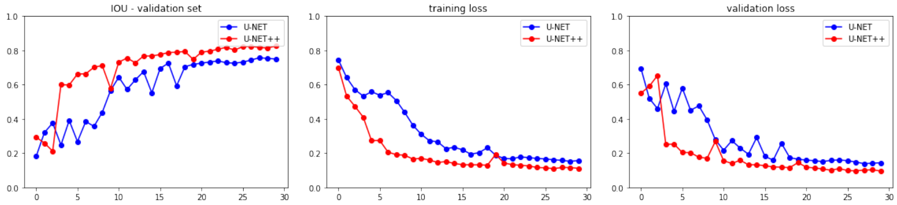

Figure 3: Training statistics using 'dice loss' metric for U-net and U-net++

Using **'Focal Loss' :**

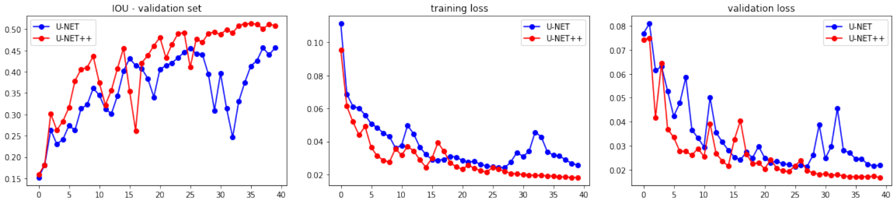

Figure 4: Training statistics using 'focal loss' metric for U-net and U-net++

5.2 Evaluation

For each of the trained models, we calculated **IoU**, **precision** & **accuracy** of pixel labeling on labeled test data set. The threshold for identifying a nuclei pixel is set to 0.5.

Network	Loss metric	IoU	precision	accuracy
U-net	dice	0.83	0.90	0.96
U-net	focal	0.46	0.75	0.97

Here are some sample nuclei segmentation from test data set.

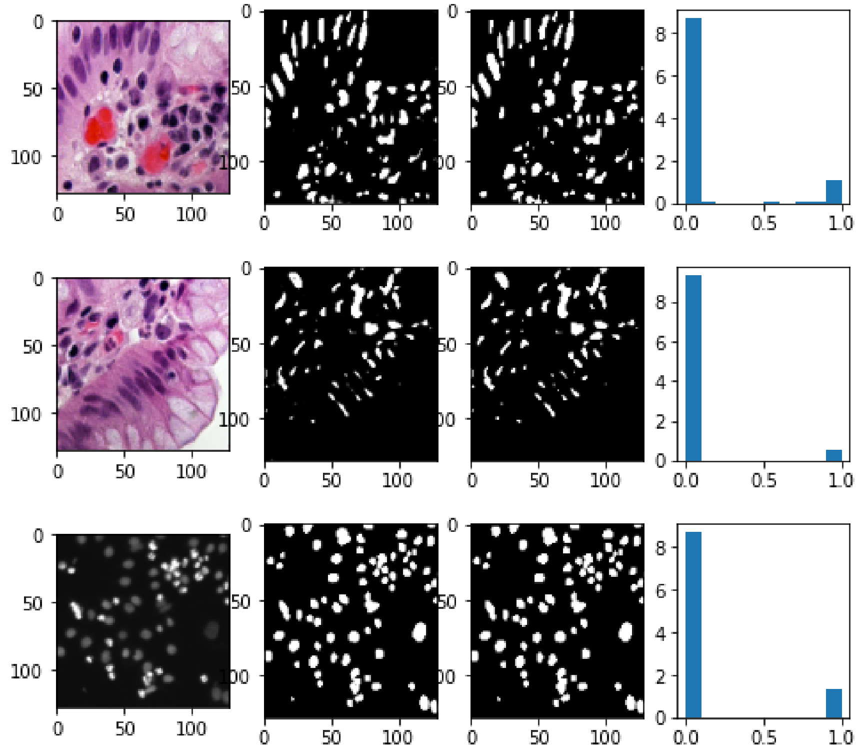

Figure 5: Segmentation by trained **U-net++** using **'dice loss'**. From left to right, each row, (a) Input image (b) probability matrix output by model (c) 0-1 segmentation with threshold 0.5 (d) distribution of pixel probabilities in model's output

These images show that the model indeed learns a non-trivial segmentation solution. Also, the distribution of pixels is mostly either 0 or 1. This suggest that the model is classifying decisively.

6 Conclusion

In this project, we have implemented and evaluated variants of U-net architecture to perform cell nuclei segmentation on 2018 Data Science bowl data set. From the results we conclude that they perform quite well on the given task. The next steps in the project could be to compare the performance with other network architectures and to use ensemble techniques for better performance.

References

[1] 2018 Data Science Bowl. Find the nuclei in divergent images to advance medical discovery. `https://www.kaggle.com/c/data-science-bowl-2018l`.

[2] 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.

[3] Feixiao Long. Microscopy cell nuclei segmentation with enhanced u-net.