Semantic Segmentation of Nuclei using Deep Networks

Submitted to Mr Bijil Prakash
as part of CS F266

Alish Dipani (2016A7PS0005G)

April 2018

1 Problem Statement

Image segmentation is one of the crucial tasks in computer vision. It has various applications like medical imaging, object detection, machine vision, face recognition, etc. . Here we attempt to create an algorithm to automate nucleus detection, for this an input image is given and output image has nuclei segmented. The input data is in the form of images of nuclei. These images are of various sizes and may contain more than one nuclei. There are 3 different kinds of slide images containing nuclei. The challenge is to be able to automatically detect the nuclei when given a particular image as input. Hence, an appropriate supervised learning model is to be trained with the data to give correct pixel label to pixels in output image to segment out nuclei. Some preprocessing of the data is also necessary. Here we have done semantic segmentation on the image to spot all the Nuclei.

2 Dataset

This dataset contains a large number of segmented nuclei images. The images were acquired under a variety of conditions and vary in the cell type, magnification, and imaging modality (brightfield vs. fluorescence). The dataset is designed to challenge an algorithm's ability to generalize across these variations. Each image is represented by an associated ImageId. Files belonging to an image are contained in a folder with this ImageId. Within this folder are two subfolders:
a. images contains the image file.
b. masks contains the segmented masks of each nucleus. This folder is only included in the training set. Each mask contains one nucleus. Masks are not allowed to overlap (no pixel belongs to two masks).
There are **3 types of slide images**. The images are of different sizes and so preprocessing is required to make them of uniform size(256x256x3). There are

total 607 images in the dataset which are further divided for training, validation and testing. For output images, the dataset contains an image per nucleus in which only the segmented nucleus is shown. Preprocessing is done to combine these images into one so as to get a combined image containing all the nuclei.

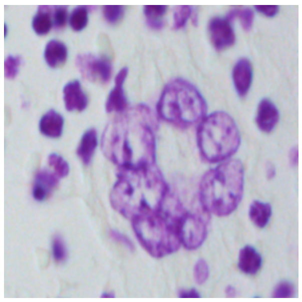


Figure 1: Type 1

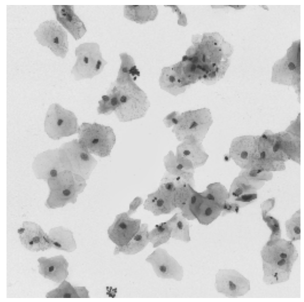


Figure 2: Type 2

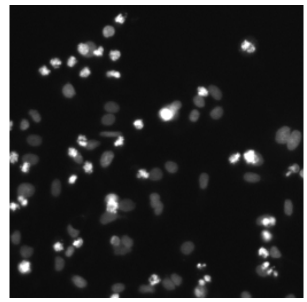


Figure 3: Type 3

3 Various Approaches for semantic segmentation

In the past, many methods have been used for image segmentation like Thresholding, Edge Detection, etc. but recently with the increasing popularity of deep learning methods many new models have been introduced which are highly accurate. These models based on Convolutional Neural Networks involve convolutional layers and up sampling or Deconvolutional layers, these models also include batch normalization to increase accuracy. Some of these models are FCN, U-net, SegNet, RefineNet, PSPNet, DeepLab v3.

4 U-net Architecture

U-net [2] is a recent model which is a based of Fully Convolutional network(FCN) [3] with a important modification. This modification is that in the up sampling part we have also a large number of feature channels, which allow the network to propagate context information to higher resolution layers. As a consequence, the expansive path is more or less symmetric to the contracting path, and yields a u-shaped architecture. The network does not have any fully connected layers and only uses the valid part of each convolution, i.e., the segmentation map only contains the pixels, for which the full context is available in the input image.

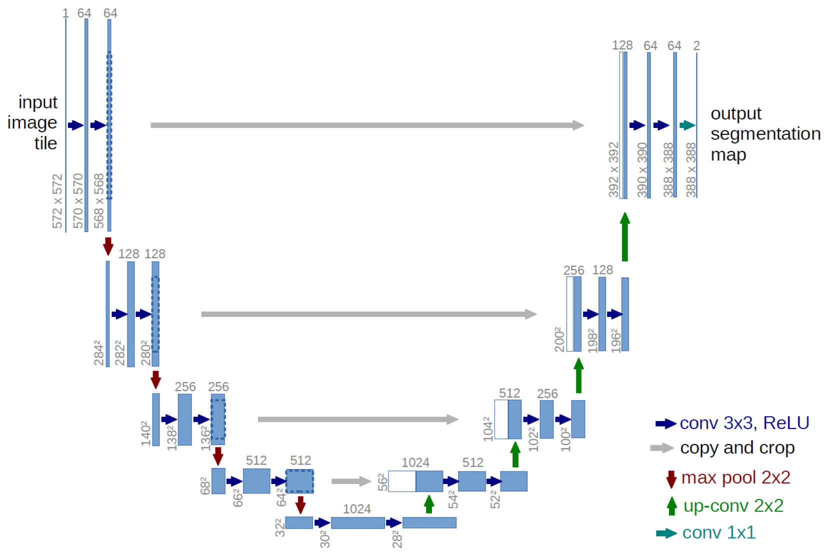


Figure 4: U-net Architecture

5 Architecture

Our Architecture is inspired by U-net and FCN contains Convolutional Layers followed by Deconvolutional layers and concatenation of outputs of convolutional layers to outputs of deconvolutional layers. Also Xavier's Initialization is done after every layer. We created the Architecture in a Low Level Framework named Pytorch [1].

The input is a 3 channel image of size 256x256 and the output is a single channel image(Grey scale image) of size 256x256. The output image has pixels corresponding to nucleus white coloured and rest of the pixels black coloured. So, the nuclei are segmented in the output image.

Examples are shown in Table 2.

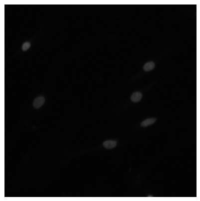
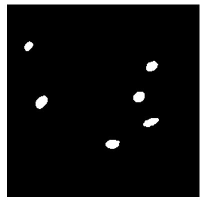
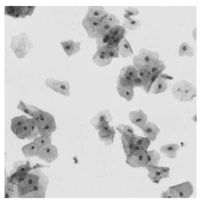
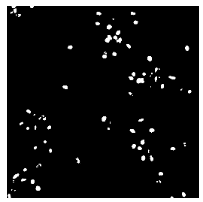
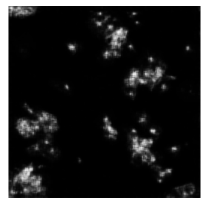
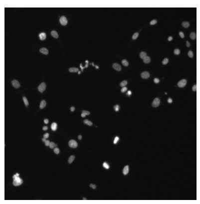
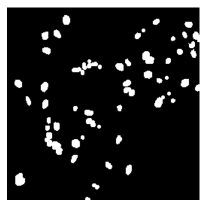
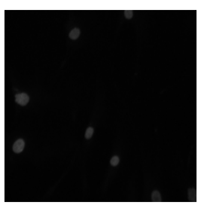
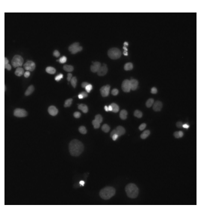
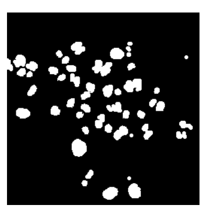
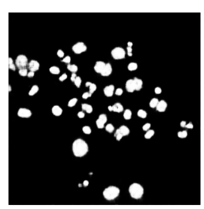
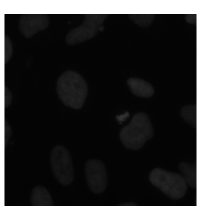
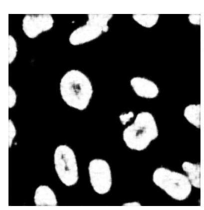
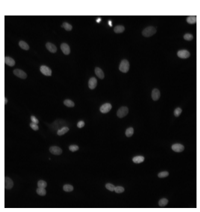
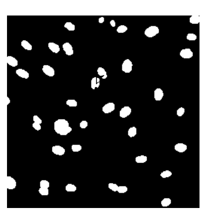
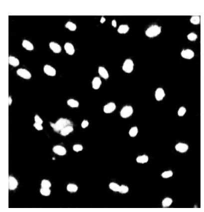
Layer	Input size	Output size	Source
2D Convolutional Layer 1	256x256x3	256x256x16	-
Xavier Initialization	-	-	-
ELU Non-linearity	-	-	-
Max Pool 1	256x256x16	128x128x16	-
2D Convolutional Layer 2	128x128x16	128x128x32	-
Xavier Initialization	-	-	-
ELU Non-linearity	-	-	-
Max Pool 2	128x128x32	64x64x32	-
2D Convolutional Layer 3	64x64x32	64x64x64	-
Xavier Initialization	-	-	-
ELU Non-linearity	-	-	-
Concat Layer 1	64x64x64	64x64x96	Max Pool 2
DeConvolution Layer 1	64x64x96	64x64x32	-
Xavier Initialization	-	-	-
ELU Non-linearity	-	-	-
Max UnPool 1	64x64x32	128x128x32	-
Concat Layer 2	128x128x32	128x128x48	Max Pool 1
DeConvolution Layer 2	128x128x48	128x128x16	-
Xavier Initialization	-	-	-
ELU Non-linearity	-	-	-
Max UnPool 2	128x128x16	256x256x16	-
Concat Layer 3	256x256x16	256x256x19	Input
DeConvolution Layer 2	256x256x19	256x256x1	-
Xavier Initialization	-	-	-
Sigmoid Non-linearity	-	-	-

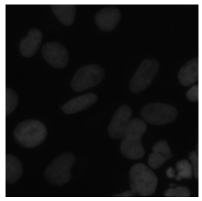
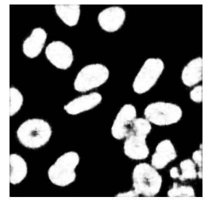
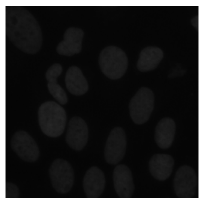
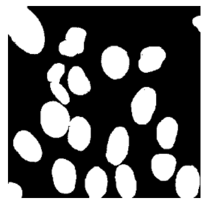
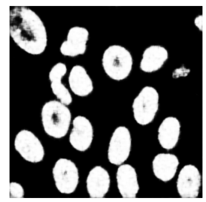
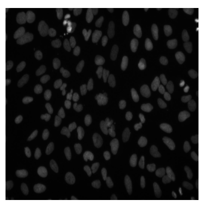
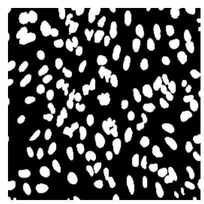
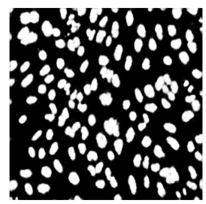
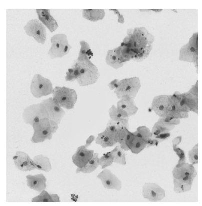
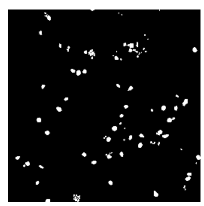
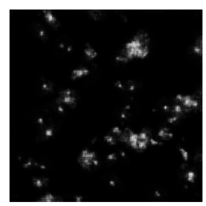
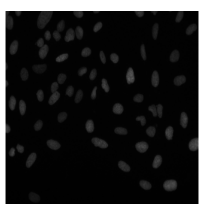
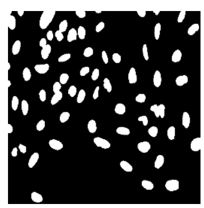
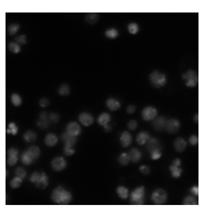
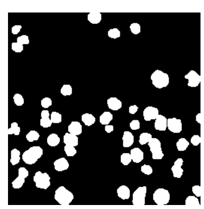
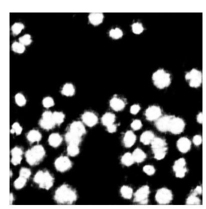
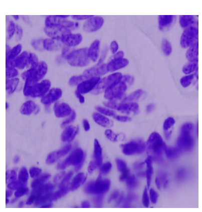
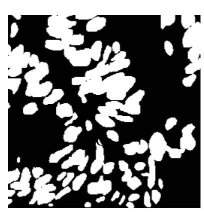
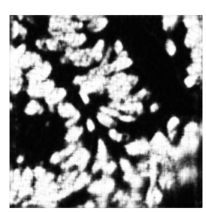
Table 1: Architecture

6 Training

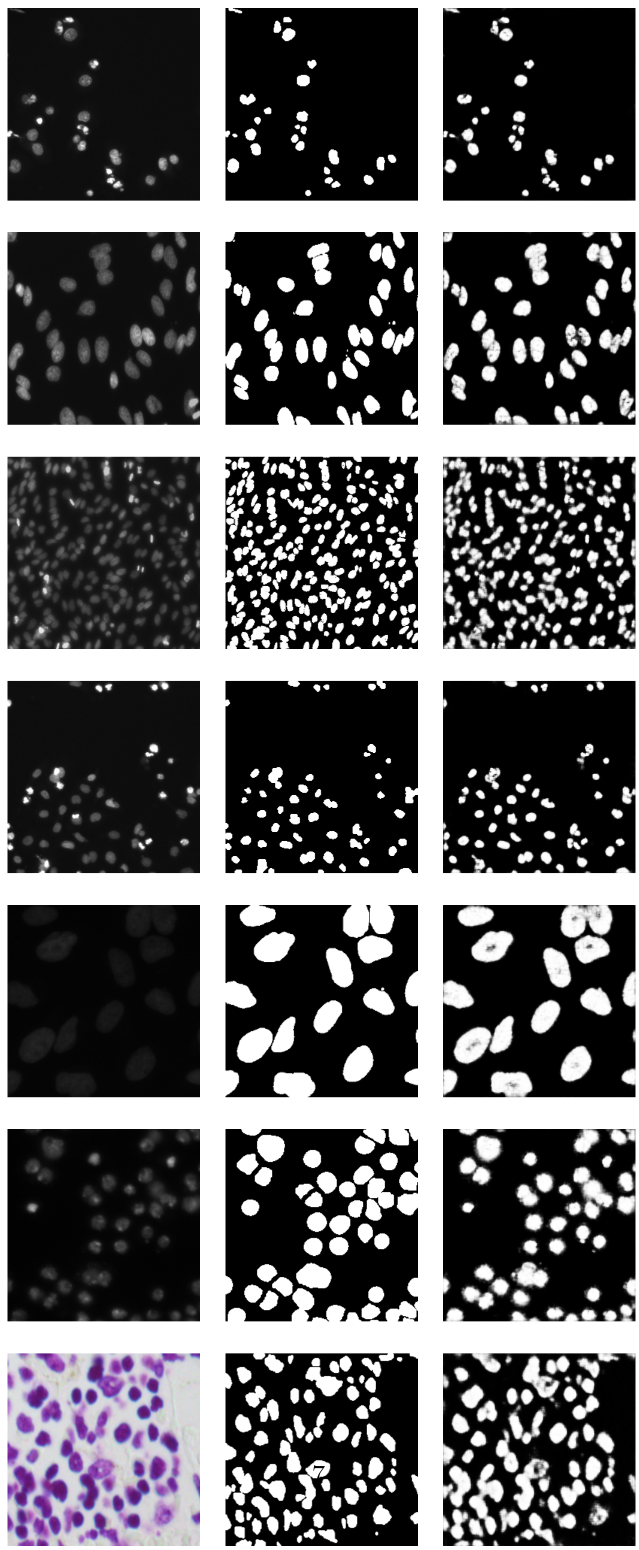
Activation used in the last layer is Sigmoid with Cross Entropy Loss.The model is trained for 50000 iterations with batch size 10, Adam optimizer is used with learning rate 0.01 and beta1 0.9 and beta2 0.999. The Learning rate decayed by factor 0f 0.1 every 10 epochs.

Table 2: Examples

Input	Annotations	Model Output
		
		
		
		
		
		
		

Input	Annotations	Model Output
		
		
		
		
		
		
		

Input	Annotations	Model Output



7 Conclusion

A lot of datapoints in the dataset were poorly annotated. Our classifier was able to segment nuclei which were not annotated in the dataset. We also created a custom architecture which used other neural networks as reference and noticed several instabilities in the network that needs to be further investigated.

8 Remarks

All the examples and the code can be found here. The semantically segmented image can be clustered for finding out various instances.

References

[1] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[2] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).

[3] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, 2017.