# U-net CNN: For finding the nuclei in divergent images to advance medical discovery

**Report in ANN and fuzzy logic system**

**By Anil jat (B1810944EE) under the guidance of Dr. shihabudheen K. V.**

## Abstract:

Using CNN to speeding up the research for diseases, from lung cancer and heart diseases to rare disorders. People's surfer from disease like heart diseases, chronic obstructive pulmonary diseases, heart diseases, Alzheimer's and diabetes. Many lives would be saved if cure comes faster.

We can take help of artificial neural network for detection of nuclei. By automating nucleus detection we can help to cure diseases faster.

## Introduction:

In recent years the deep convolutional networks outperformed the state of the art in many visual recognition tasks. But the convolutional networks have already existed for a long time ago, but that time enough datasets was not available. The breakthrough by Krizhesky et was due to supervised training of a large network with 8 layers and millions of parameters on the ImageNet datasets with 1 million training images.

Generally the use of convolutional neural networks is on classification task. However in many visual tasks especially in biomedical image processing, the desired output supposed to be assigned to each pixel.

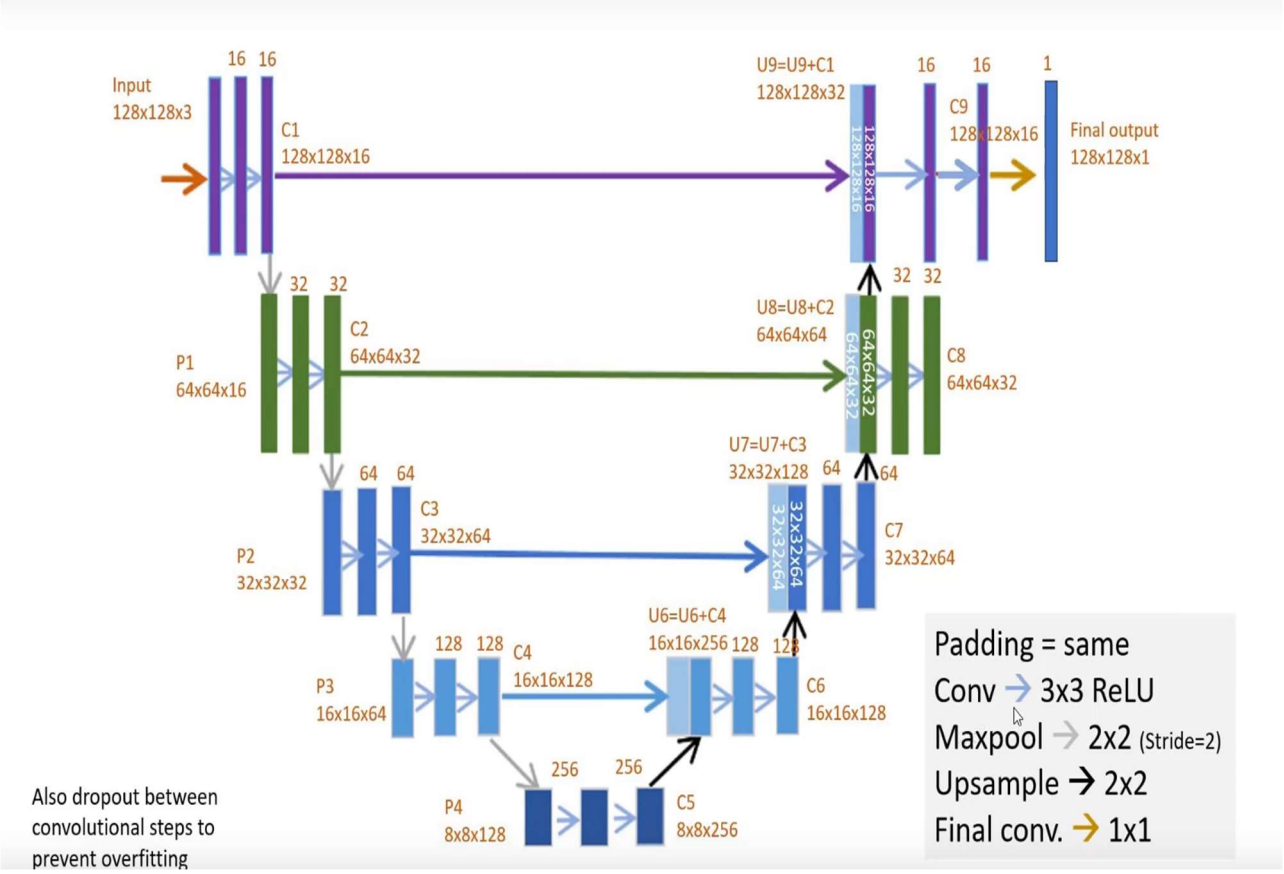
The main idea is to implement a usual contracting network by successive layers, where pooling operators are replaced by up sampling operators. Hence, these layers increases the resolution of the output. In order to localize, high resolution features from the contracting path are combined with the up sampled output. A successive convolution layer can then learn to assemble a more precise output based on this information.

One important modification in this architecture (U-net) because of biomedical image segmentation is that in the up sampling part we have also 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 contracting path, and yields a u-shaped architecture. The network does not have fully connected layers and only uses the valid part of each convolution, i.e. segmentation map only contains pixels, for which the full context is available in the input images. This strategy allows the segmentation of arbitrary large images by an overlap-tile strategy. To predict pixels in the border region of the images, the missing context is extrapolated by mirroring the input images. This tiling strategy is imported to apply the network to large images, since otherwise the resolution would be limited by the GPU memory.

## Network architecture (U-Net):

It consist of contracting path(left) and expansive path(right) or we can also say encoder and decoder. The contracting path is same as a typical architecture of convolution network with repeated application of 3X3, ReLu and a 2x2 max pool operation. At each downsampling we double the number of channels. In expansive path each step is consist of upsampling of feature map followed by 2x2 convolution that halves the number of channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by ReLu.

The cropping is necessary due to the loss of border pixels in every convolution. At the final layer 1x1 conv is used to map component feature vector to the desired number of classes.




## Data:

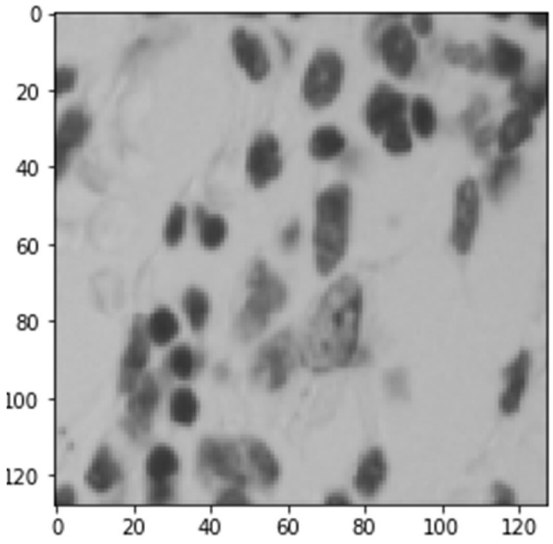



*Figure 1. Real image*

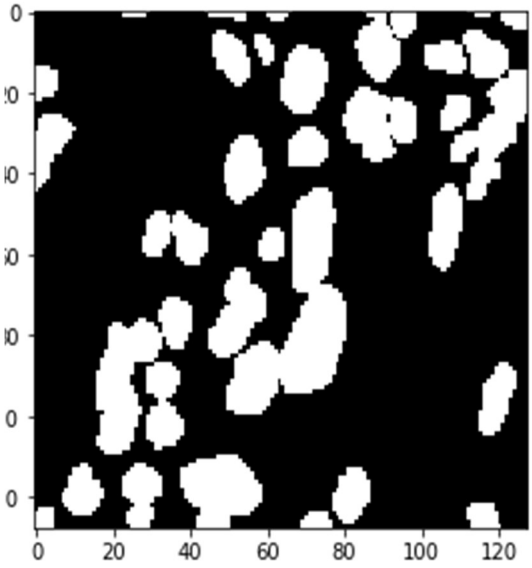


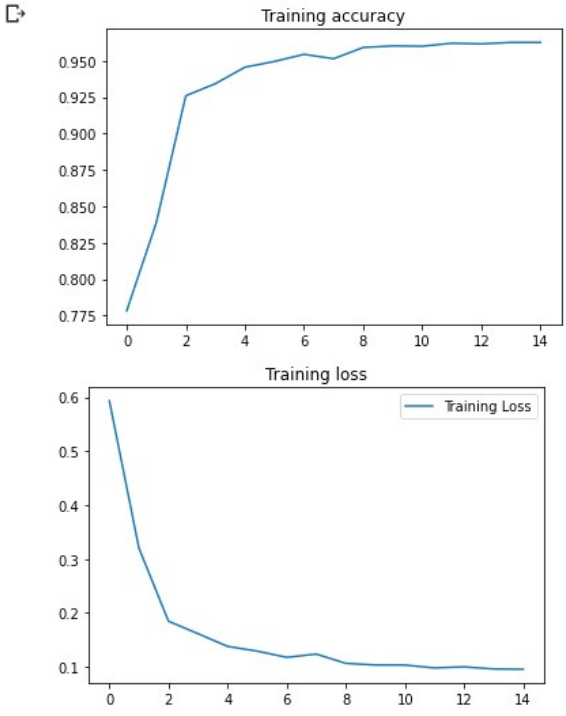

*Figure 2 predicted nuclei in input image*

## Training:

+ Code + Text

```
acc = results.history['accuracy']
loss = results.history['loss']
epochs = range(len(acc))
plt.plot(epochs, acc, label='Training accuracy')
plt.title('Training accuracy')
plt.figure()
plt.plot(epochs, loss,label='Training Loss')
plt.title('Training loss')
plt.legend()

plt.show()
```



## Final predictions:


## Resources:

1. https://www.kaggle.com/c/data-science-bowl-2018/overview
2. For data pre processing
   https://github.com/bnsreenu/python_for_microscopists