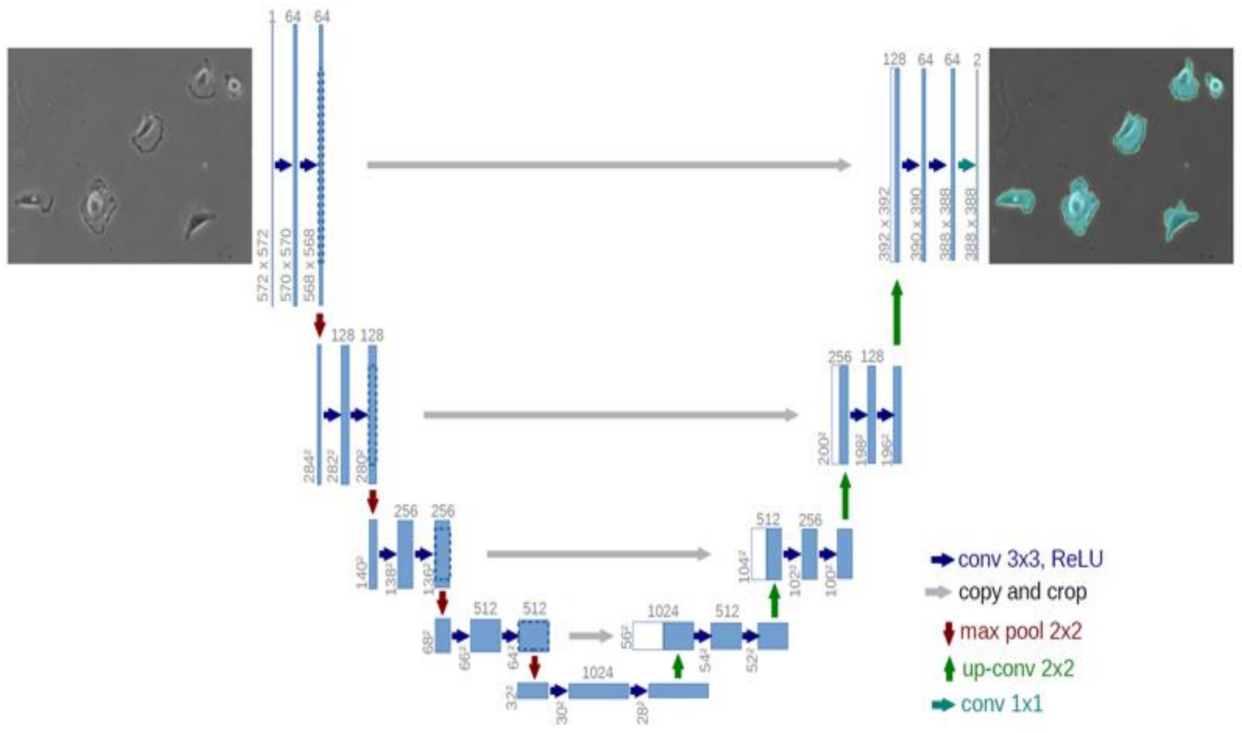
Nuclei segmentation in H&E lung slide pathology images.

Background:

There have been lot of traditional segmentation algorithms available for the segmentation of nuclei in pathology images but the latest advancement in deep learning networks had proven that using CNN has more efficacy than the old traditional implementation. Hence CNN-UNet is utilized to implement the model.



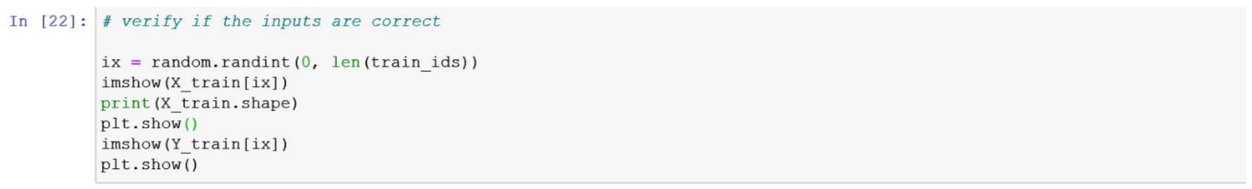
Procedure:

- Initially two pathology of H&E lung slide (diagnostic) images are downloaded from TCGA repository.
- Three ROI’s with different nuclei variations are selected in each image.
- Tiles are extracted from the respective ROI’s.
- Nuclei masks are generated using nucleus parameters and those images are used as ground truths
- All the above steps are implemented using QuPath.
- These images are later converted to gray scale.

- CNN - Unet model using the gray scale images is implemented.

Code Analysis:

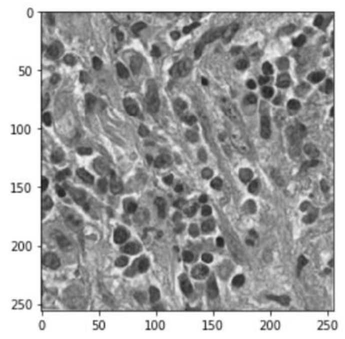
- All the required imports are imported, mainly the model was built using keras with tensorflow backend.
- All the RGB images are converted to grayscale and saved into a folder for further usage.
- Data augmentation is performed using keras data generator and augmenter to increase the size of data set with augmentations such as horizontal flip, vertical flip, rotation, zoom.
- Now the data set has increased from 30 images to 150 images which are later fed to the network.
- All train, mask, test Images are resized to 256*256 since neural networks receive inputs of the same size, all images need to be resized to a fixed size before inputting them to the CNN.
- Now both the train and mask images are verified by providing randomly generated index to check if the input to CNN is correct.

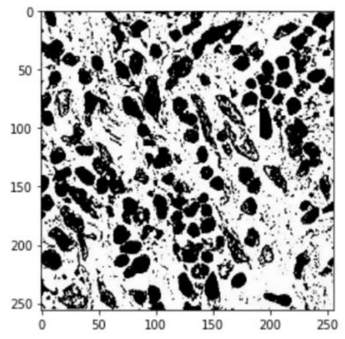


```
In [22]: # verify if the inputs are correct

         ix = random.randint(0, len(train_ids))
         imshow(X_train[ix])
         print(X_train.shape)
         plt.show()
         imshow(Y_train[ix])
         plt.show()
```

```
(150, 256, 256)
```





Model:

- Initially the model was built and verified using the source code from github repo’s with identical research of nuclei segmentation, but the loss was really high using those models because of the bias and under fitting. The below mentioned layers were added to increase the efficiency of the model which reduced the loss to some extent but not totally.
- Activation function is changed to ‘relu’ for inputs, optimizer is changed to ‘sgd’, Binary cross entropy as loss function, activation for output is changed to ‘softmax’, various epochs and batch sizes has been experimented.

c6 = Conv2D(512, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same') (p5)

c6 = Dropout(0.3) (c6)

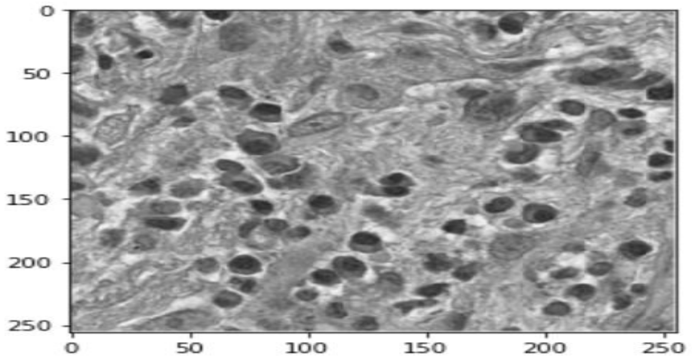
c6 = Conv2D(512, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same') (c6)

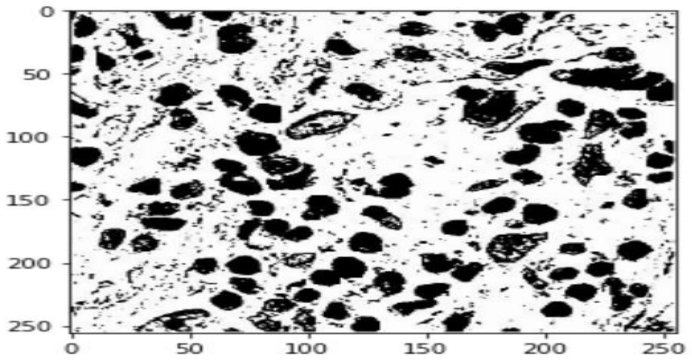
p6 = MaxPooling2D(pool_size=(2, 2)) (c6)

```
c11 = Conv2D(1024, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same') (p6)
c11 = Dropout(0.3) (c11)
c11 = Conv2D(1024, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same') (c11)

u12 = Conv2DTranspose(512, (2, 2), strides=(2, 2), padding='same') (c11)
u12 = concatenate([u12, c6])
c12 = Conv2D(512, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same') (u12)
c12 = Dropout(0.2) (c12)
c12 = Conv2D(512, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same') (c12)
```

- Now the model is predicted using the test images, certainly a lot of room for improvement because the loss is really high and the limited data set size.





Future Work:

Few images are tested and the predicted image is sometimes black, hence the future work would be to increase the model efficacy by training with more images and decrease the loss.