COMP 540 Final Project

Chengyin Liu & Ran Jin

RICE Unconventional Wisdom

Project Pipeline

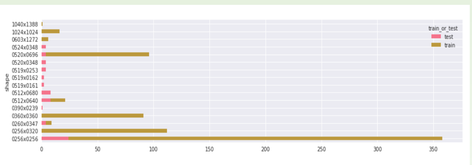
Motivation → Exploratory Data Analysis → U-net & Mask RCNN fitting & Selection → Leaderboard Results → Our Takeaways

Motivation

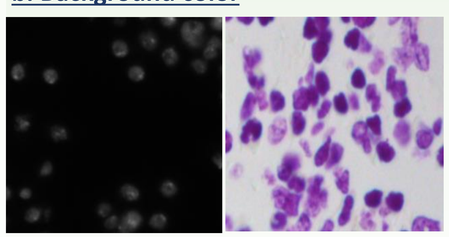
A lot of people are suffering from all various diseases like cancer, etc. Some of the diseases are still curable within a certain time constraint (i.e. if the nuclei can be precisely detected in the early stage). The purpose of this Data Science Bowl is to detect nuclei from cells faster in a more accurate manner. Specifically, we want to develop a generalized algorithm that does the detection for us in automation. Thus, overall this competition is a great practice for us to apply what we have learned in class to a generalized segmentation problem.

Exploratory Data Analysis

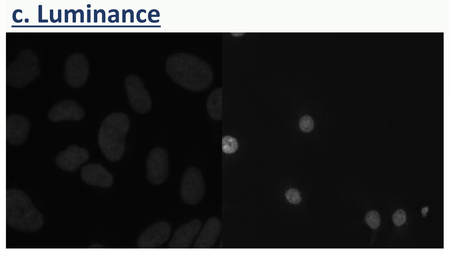
a. Image size



b. Background color



c. Luminance



d. Non-nuclei that look like nuclei

U-net

Why U-net?

The architecture of this model is consisted of two paths. As we input the images, it captures the structure and contracts it down to the bottom by successive layers, where pooling operators are replaced by unsampling operators in order to increase the resolution. Thus, by combining the high resolution features with unsampled operators, it enables precise localization precisely to propagate context information to higher resolution. It was showed that such network can be well trained from an end to end with few images which is our case here.

Parameter combinations:

1. Learning rate: 0.001, 0.0001 (Adam)
2. Starting depth of the network: 8, 16, 32, 64.
3. Image size: 128*128, 256*256, 384*384, 512*512
4. Batch size: 8, 16
5. Number of epochs: 30, 50.
6. Filter size: (7,7), (3,3), (2,2)

The numbers written in red is the best combination (corresponds to highest accuracy).

Additional stuff:

1. Added dropout after each convolutional layer to control over-fitting.

Reference:

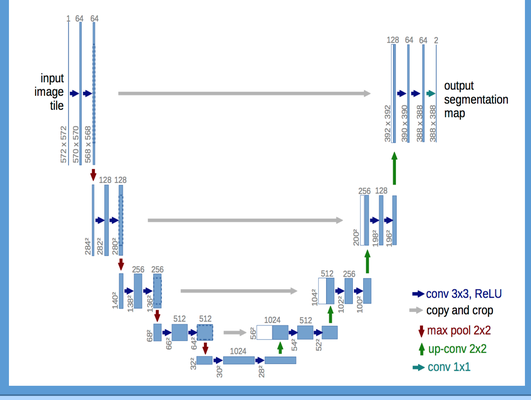
https://arxiv.org/pdf/1505.04597.pdf

https://www.kaggle.com/keegil/keras-u-net-starter-lb-0-277?scriptVersionId=2164855

https://www.kaggle.com/raoulma/nuclei-dsb-2018-tensorflow-u-net-score-0-352

Building Predictive Model

Train the model in training/validation set → Validate the parameter in training/validation set → Test the model in testing set → Combine the results from different models

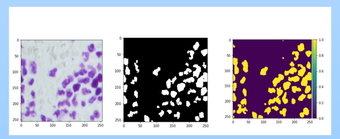


Problems encountered:

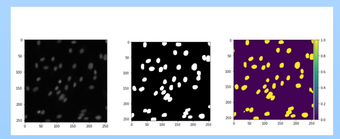
1. As most of other might have, since running the code on our laptops are extremely slow, we decided to use Goog Colab to run the code.
2. For stage one, since the testing data was small, running the code was only about 2 hours or so. However, stage 2 had over 3000 images for test dataset. And Google Colab had a hard time reading all the images (Putting them all in an array). Thus, we had to separate all the test images into 20 zipped files and separately run them to get 20 csv. Files and then put them together in one .csv file.

Result:
Stage 1:
Baseline: 0.252
Best: 0.324
Stage 2:
Final: 0.422

Training data fit using our developed model



Validation data fit using our developed model



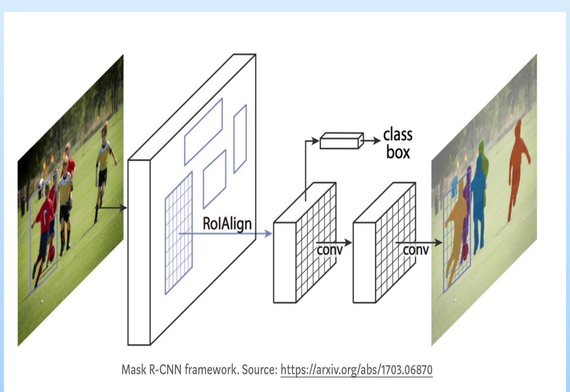
Mask RCNN

Overview:

Mask R-CNN improves the Faster R-CNN by adding a third branch that gives the output of the object mask from Faster R-CNN. It has two stages. The first stage scans the image to generate proposals (Region Proposal Network - RPN), and the second stage classifies the proposals to generate bounding boxes and masks as we see in the image in the middle.

It is consisted of three modules:

1. Backbone
   - ❶ Region Proposal Network – RPN
   - ❷ Feature Pyramid Network (FPN)
2. ROI Classifier & Bounding Box Regressor
3. Segmentation Masks



Mask R-CNN framework. Source: https://arxiv.org/abs/1703.06870

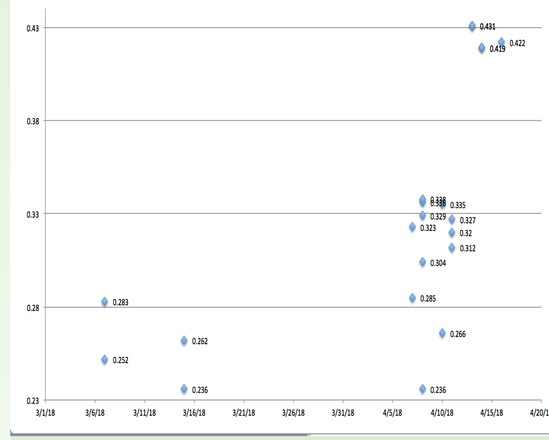
Result:
Stage 1:
None
Stage 2:
Baseline: 0.419 (Resizing image to 256*256)
Final: 0.431 (Resizing image to 512*512)

Reference:

https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46

Leaderboard Results

We started submitting our results on 3/7/2018. We had a total of 21 submissions. As you can see, we have improved from 0.252 to 0.431 for the overall accuracy on the leaderboard. The highest score for Mask RCNN was 0.431, and the highest score for U-net was 0.422. Overall, we were ranked 417/3634 (Top 12%) upon the completion of the competition. The following is the accuracy plot against dates submitted:



The points before 4/11/18 were using stage 1 testing and stage 1 training data.
The points after 4/11/18 were using stage 2 testing and stage 1 training data.
In stage 1, our highest precision score was 0.338.
In stage 2, our highest precision score was 0.431
Several reasons that our accuracy has increased in a short time period:

- More model selections (Added Mask RCNN to our plate)
- Increase the size of compressed images
- Increase the learning rate or epoch numbers

Our takeaways

- EDA is extremely important for us to understand the data, also how to specifically approach the problem using the data available, and how to turn those data into an executable phase.
- As validation loss goes down, it does not necessarily mean that the accuracy would go up.
- For every run, even with the same parameters and model, the accuracy could vary probability within + or – 0.05 accuracy range.
- Resizing images is easier for training but it does make some of the images lose information in the sense that if an image of 256*256 was downsized to 128*128, some information of the original image is lost. Thus, this is a trade off between complexity of training and information presented in the images.
- Deeper Neural Networks capture more model structure than the shallower ones.
- Algorithms like U-net are quite complicated in terms of number of layers and depth which would cause really long training time if executed on our own laptops. Thus, a lot of times we would have to rely on other services like AWS or Google Colab that significantly optimizes the running time.