Learning Cell Counts

Cody Kesler

February 18, 2020

1 Introduction

A Bio-Chem Lab at BYU campus is experimenting with a treatment created by the Washington University for a side effect of Chemotherapy which affects the lungs. 3 groups of mice, a control group, and two infected groups one with treatment and one without. A sample is scraped from the lungs of each mice from each group.

2 Problem

A cell sample is scraped from the lungs of each mouse from each group. The sample is then looked at under a microscope and 3 different types of cells are manually counted. The ratio of the different cells that appear signals how infected the lungs are, and therefore how well the treatment works. This counting of cells is a long manual process and time constraints limit the number of cells that can be looked at from one sample.

Automating this task will help increase the number of cells that can be looked at per sample, increase the accuracy of each cell type count, and reduce the time spent looking to see how successful the treatment was. The problem then is to train a DNN which will automate the task of counting the number of 3 different cells that appear in a microscopic image.

3 Data

The data for this project will be supplied by the Bio-Chem lab on campus from the images of the cell slides. Currently, the data set is fairly small with 30-40 control samples, and 14 infected samples with only 5 of them with the treatment. The images are 4000x6000 each. As the lab hadn't labeled the data, only recorded the counted ratio of the cells, labeling and time constraints had to be taken into account. This led to a total of around 90 images being completely annotated with a mask for where each type of cell is.

4 Approach

The approach taken to tackle this problem ended up being a two step process. Use deep learning to find the location of each type of cell then use computer vision blob counting technique to count the number of cells. The deep learning approach evolved as the data and problem were analyzed.

4.1 The Proposal

When originally researching the topic object detection was in the forefront of the ideas to be experimented with. The proposal thus dealt with finding state of the art methods including YOLO (You Only Look Once) and trying autoencoders to discover the latent space of cell distributions pre type of cell. As both of these ideas were a place to start neither panned out once more information about the problem and the data were obtained.

4.2 Pretraining and Fine Tuning

As the understanding of the data took place, it became clear that a network which would produce a mask of the cells would be the proper way to handle the data. A U-Net Architecture was implemented to train on the images and map them to their corresponding masks. Concern of the small training dataset led to the thought process of pretraining the data on a similar dataset consisting of cells or nuclei. The image dataset from the 2018 Data Science Bowl, was used as the pretraining dataset. The U-Net learned the training set well, but completely failed when used on the BYU-Mice dataset. When training the image returned from the network was almost always completely masked(blacked) out, opposite of what was believed to have happened. After training for 20 epochs, no reasonable results appeared, which led to the investigation other avenues for results

4.3 Finally, (Limited) Results

With the pretraining of the network a failure, another approach had to be tried. The obviously simple answer was to train the U-Net from scratch. The approach of training a model for each type of cell with a custom loss function was implemented. The proposed loss function is as follows:

$$L_{cell=i}(\hat{y}, y_i, \overline{y}_i) = ReLU(\alpha * CrossEntropyLoss(\hat{y}, y_i) - \beta * CrossEntropyLoss(y, \overline{y}_i) + margin) \quad (1)$$

where y_i is the target mask i and $\overline{y}_i$ is the combination of the masks of the other cells: $\Sigma_{j!=i} mask_j$. Values of $\alpha = 2$ and $\beta = .1$ were used for these experiments.

This loss encourages the network to produce masks similar to the target mask and punishes it for being close to the mask of the other type of cells. Training separate models with custom loss functions provides a first attempt to avoid needing the model to generalize well across the different types of cells. This method provided a

5 The U-Net and Counting Blobs

The U-Net Structure is very similar to the structure of [1]. The images are resized to 256x256 and fed through a down sizing set of convolutions. The images are then upsized with skip layers between each corresponding downsizing layer to preserve spacial knowledge in the model. The images are then fed through a computer vision algorithm which counts the number of individual blobs in the image. The images shown below are a cell masks for an image for the two types of cell which are being counted.

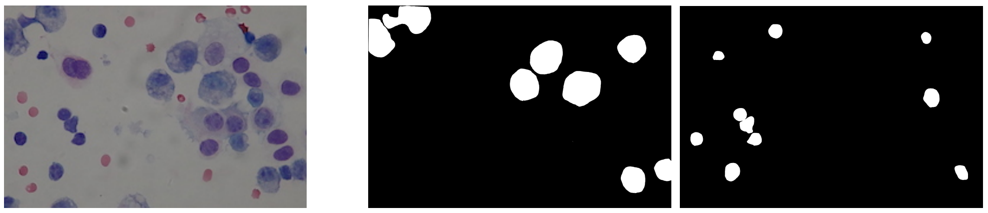
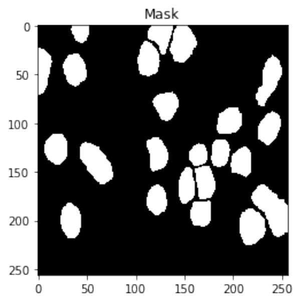

(a) Original (b) Mask for Macrophages (c) Mask for Lymphocytes

Figure 1: Untreated Mouse Lung Cells

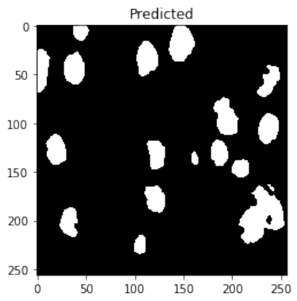
6 Results

As time and data were limited the results of this experiment were also limited.

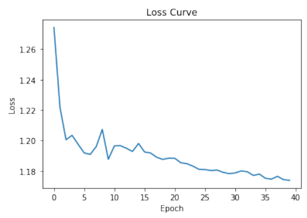
Training on the Macrophages masks was fairly successful in terms of finding the clumps of cells. One issue the model encountered was learning the small separations between some of the cells to count them as distinct.



(a) Macrophages Mask



(b) Predicted Macrophages



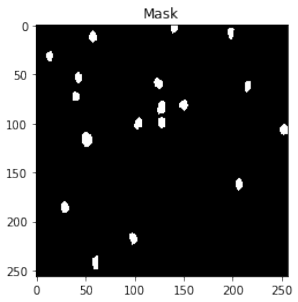
(c) Macrophages Loss

Figure 2: Macrophages

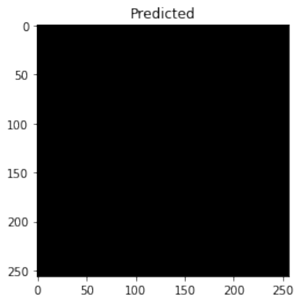
Training on the Lymphocytes was unfortunately less fruitful. The model was not able to predict any lymphocyte cells and the loss converged fairly fast. This could have been caused my a number of things. One cause is that the there is not enough lymphocyte cells per image to give the model incentive to move toward the goal, especially with a blacked out mask is ”far away” from the macrophage mask.

This could be fixed by increasing the α and lowering the β terms. This tweak unfortunately also did not improve or change results, even with β set to 0.

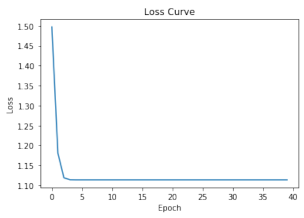
Another approach to fix the issue is to take the model trained on the macrophages and train it on



(a) Lymphocytes Mask



(b) Lymphocytes Macrophages

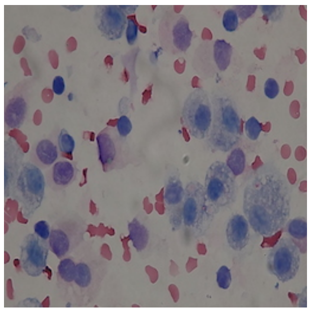


(c) Lymphocytes Loss

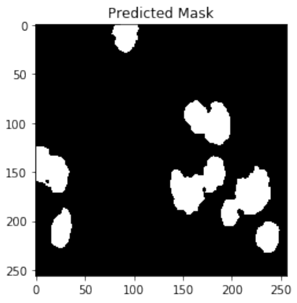
Figure 3: Lymphocytes

7 Future Work

As only training success was obtained with the Macrophages with the current technical approach we will test the Model on recognizing and counting macrophages. As seen in figure 4 the model was able to pick out macrophages in the image at a high level. In order for the model to learn the nuances of separating bunched cells in order to count correctly more data and a more fine tuned loss algorithm to punish large clusters, encouraging separating the mask of each cell. To encourage the model to learn the lymphocytes the loss function also needs to be tweaked to encourage marking cells on a mask.



(a) True Image



(b) Predicted Mask

Figure 4: Lymphocytes

References

[1] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. 2015.