U-Net On Biomedical Images

Amir Persekian aperseki@eng.ucsd.edu
Max Jiao mjiao@eng.ucsd.edu
Lucas Tindall ltindall@eng.ucsd.edu

Abstract

Traditional CNN architectures have been very good at tasks such as applying a class label to an image after training on a very large example dataset. However, in many applications, a more fine grained classification is necessary. One example is for image segmentation in the biomedical field. In order to classify pixels based on only the contextual information surrounding it without a need for a large training dataset, we use an efficient and generalizable network called the U-Net. The U-network leverages the homogeneous nature of medical images to train on similar images very efficiently. Our network is able to classify with an F1 score of 0.96 after training on just 10% of our images, but loses precision around the edges of of said structures.

1 Introduction

Image segmentation and pixel-wise classification are an important next step when it comes to image processing techniques. Current architectures are very good at classifying regions or images based on sliding windows. There are some problems when it comes to the traditional approach. These problems can be seen and emphasized in the field of biomedical research and imaging. Traditional CNN models rely on a vast amount of training data, primarily relying on feature extraction and feature matching in order to correctly classify images or regions of images. Convolution and other techniques are used for increasing scaling and translation invariance, but the primary motive has been to extract and match features.

In the biomedical field, classification tasks are generally less about detecting what an image is, but rather identifying the boundaries between different components and detecting where key sections of the specimen are located. Traditionally, this is done by staining specimens and imaging them one at a time. Scientists proceed to hand trace and label the different parts of the images based on their domain knowledge. This is incredibly arduous and has a very low throughput, which is a bottleneck in being able to do larger scale research such as time series observations on specimen in response to different stimuli, or efficient 3D rendering and extrapolation.

The goal is to have a network accurately label related images using a very limited training set. This means that a greater importance is put on data preprocessing techniques as well as contextual information in the images. Thankfully, images in this field tend to be homogeneous not only between images, but even across regions in the same image.

In order to realize these goals, we start with three different tasks using different datasets containing microscopic cell images. We use the architecture proposed by O. Ronneberger [1], known as the U-Net. The paper states that this architecture is both robust and generalizable to applications in the biomedical field. The tasks that we attempt in this project are:

- Identifying nuclei in different images of cells
- Identifying cell membranes and component boundaries
- Differentiating different cell components in images of cells

2 Related Work

2.1 U-Net

Convolutional neural networks have become very common tools for image processing tasks. Their ability to learn detailed spatial features makes them well suited to the task of image segmentation since they have the ability to detect features invariant of location or orientation. A common architecture used for the specific task of biomedical image segmentation is the U-net model first proposed by Ronneberger et al [1]. This architecture is well suited to the task of pixel-wise classification as the network is fully convolutional with outputs being the same size as the inputs.

2.2 Data Augmentation

Given the limited and small datasets available, there is a strong need for data augmentation to boost the variance of the datasets. With stacks of images often resulting in very homogeneous features data augmentation allows the network to learn a better representation of the feature space. Multiple works [4, 5] have also used random crops, flips and rotations to bolster their datasets.

2.3 Other Architectures

Since there are many biomedical image segmentation datasets there have been many proposed methods and architectures to best solve the task. Some very similar architectures used on these same datasets include the Residual U-Net [5] and Mask R-CNN [6]. The Residual U-Net combines the typical U-Net architecture with residual convolutional blocks for increased performance. For our purposes and because of our limited computational resources we choose to forgo this model and keep the original U-Net architecture. Mask R-CNN has also shown to perform well on this task but requires a region proposal network which adds additional layers of computation.

3 Dataset and Features

For testing our architecture, we used a variety of different datasets in order to test the three major goals that we stated earlier. The goal was to use as small of a dataset as we could for training, and keep the rest for validation. Initially, we used 80% of the dataset for training, and then 20% as validation, but then we gradually decreased the amount needed for training down to 10%.

Set 1: Kaggle Data Science Bowl 2018 Dataset [7] First is the Kaggle 2018 dataset, which features a variety of types of images with nuclei in them. The goal we pursued on this dataset was to identify the nuclei of the cells, which is the first step in detecting separate key objects in an image from the biomedical field. The challenge on this dataset was that there are many different types of images that were taken using different magnifications and stains.

Kaggle: Details The original dataset contained 670 color images of various pixel dimensions. These were all downsized to a resolution of 128 x 128 pixels.

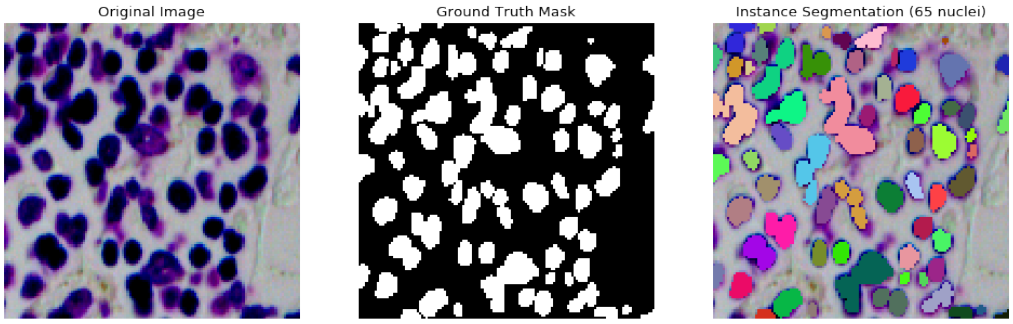

Figure 1: Sample image, ground truth mask, and instance segmentation from Kaggle dataset

Set 2-3: ISBI Datasets [8, 9] We used two separate datasets from the ISBI challenges. These were the 2012 and 2013 segmentation challenges that gave the task of segmenting neuron structures in Electron Microscope (EM) images. These images are devoid of color and pretty complex. However, they are very similar across the dataset. This is especially true of the 2013 challenge where the dataset consisted of stacks of cross sections of neurons. The target results were masks that clearly marked out the boundaries of the neurite components.

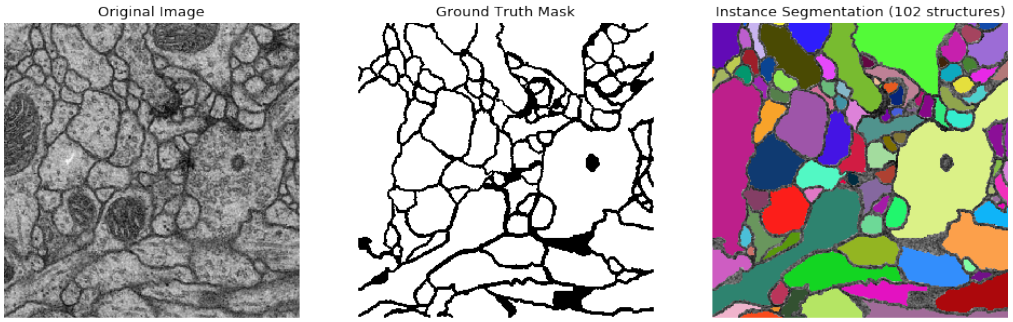

Figure 2: Sample image, ground truth mask, and instance segmentation from ISBI 2012 dataset

ISBI: Details The ISBI 2012 dataset had 30 binary images of resolution 512 x 512 pixels. The ISBI 2013 dataset had 100 binary images of resolution 1024 x 1024 pixels. These images were downsized to 512 x 512 pixels.

Set 4: NCMIR Mitochondria Dataset [10] The NCMIR dataset was collected using an Electron Microscope (EM) and contained images from the brain of a mouse. The images were annotated such that each mitochondria was identified. This touches on our 3rd goal of identifying specific parts of a cell in an image.

NCMIR: Details This dataset contained 30 binary image/mask pairs of resolution 1024 x 1024 pixels. These were downsized to a resolution of 512 x 512 pixels.

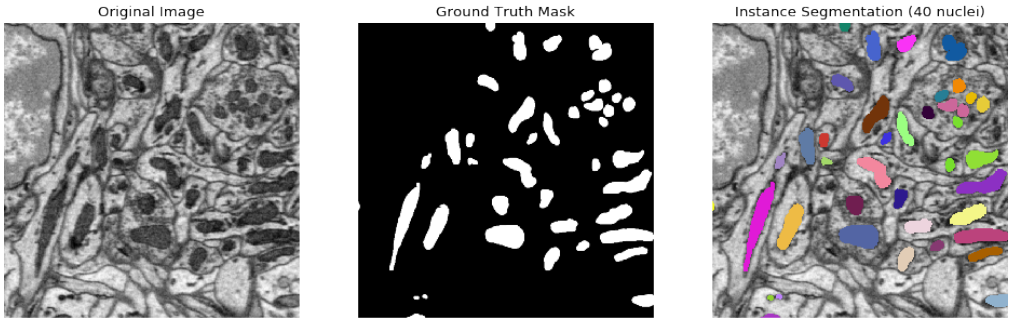

Figure 3: Sample image, ground truth mask, and instance segmentation from the NCMIR dataset

Set 5: EPFL CVLab Mitochondria Dataset [11] The EPFL dataset was another EM collection which contained a stack of images taken from the CA1 hippocampus region of the brain. The masks identified mitochondria within the images.

EPFL: Details The dataset contained 165 binary image/mask pairs of resolution 768 x 1024 pixels. These were downsized to a resolution of 512 x 512 pixels.

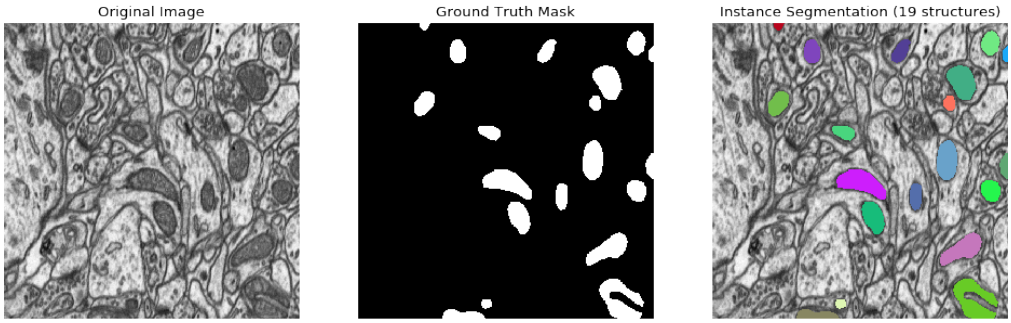

Figure 4: Sample image, ground truth mask, and instance segmentation from the EPFL dataset

Pre-processing and Post-processing One very important consideration to our posed problem is that training data need not be readily available. Many other architectures may be able to do the segmentation and classification jobs very well, but the downside is that they require a large amount of training data before results become convincing. In order to address this issue, we leverage certain quirks of these microscopy images to train our model successfully despite the limitations. First, because the images are always very similar to each other when taken with the same machine on the same types of cell, images don't vary very much across the dataset. This means that we can train on context regions within the image using the U-Network, which does not have any fully connected layers. This further extends into capabilities to handle any reasonably large sized image, and in fact treat the greater size as more training data. The second method that we use is data augmentation. It can be noted that the images across the dataset can be expressed as very similar to distortions and transformations done on each other. Therefore, we an create a sizable amount of augmented data with various image transformation techniques such as blurring, Gaussian noise, affine transformations, flips, rotations, and zooms.

There were no mandatory post-processing techniques needed to achieve the targets that were specified in the competitions that each dataset was for. However, we did decide to take our results a step further and try to further separate the different component masks that we ended up with after

feeding inputs through our model. The methods that we used in order to accomplish this will be further addressed in the methods section.

4 Methods

4.1 Preprocessing

Because of the nature of the U-Network and the specifications of the field of application of our project, we are able to limit the need for preprocessing. As previously stated, the U-Network is generalizable across any reasonably sized input image (approximately 64x64 pixels or greater), and puts an emphasis on using contextual information to classify the pixels in a region. The versatility of our input means that we do not need to down-sample, blur, or do any other augmentation techniques to improve our model's ability to classify inputs. However, because each region of the input can be considered a new training sample, larger and higher resolution inputs greatly increase the performance of our model. Since the input images are generally so similar in both composition and structure, we simulate an increased dataset using data augmentation techniques. We create a large supplementary dataset using various distortion transforms, including rotations, zooms, affine transforms, crops, and warps.

4.2 Architecture

For our network architecture we choose the U-Net proposed by Ronneberger et al [1]. The entire network can be found in figure 5. The first half of the network is an auto-encoder comprised of convolutional blocks with batch normalization and ReLU activations. These blocks are connected by max pooling layers. Since each max pooling layers decreases the spatial dimensions of the features we also increase the number of feature channels as the network progresses.

$$ReLU(x) = max(0, x) \tag{1}$$

The second half of the network is a decoder which takes in the detailed, localized features learned by the auto-encoder and combines them with skip connections and transposed convolutional layers to recover spacial features in larger resolutions. The decoder half continues with this same pattern of skip connections and transposed convolutional layers until the original resolution is recovered and then applies a final sigmoid activation, resulting in a pixel wise binary classification.

$$Sigmoid(x) = \frac{1}{1 + exp(-x)} \tag{2}$$

The U-Net was trained using stochastic gradient descent with the Adam optimizer. Since each task involved binary pixel-wise classification we used the binary cross entropy loss function:

$$BCE\,loss = -(y * log(p) + (1 - y) * log(1 - p)) \tag{3}$$

4.3 Post-processing

The target results of our model were just to classify pixels as part of the mask or part of the background. We tried to improve on this result by post-processing the result such that each individual component is distinguishable from the others. To do this, we used the K-means algorithm to find the center points of each cell or component. To determine the optimal K-value, we did a sweep of K-values with the elbow method using the Sum-Squared Error to approximate the number of clusters in the generated mask. The centers of the k-means algorithm were used as markers for the Watershed algorithm to distinguish the components from each other.

4.4 Objectives and Metrics

To accurately measure the performance of our models we evaluated both the network on the training and validation datasets using multiple evaluation metrics. First we evaluated the performance using

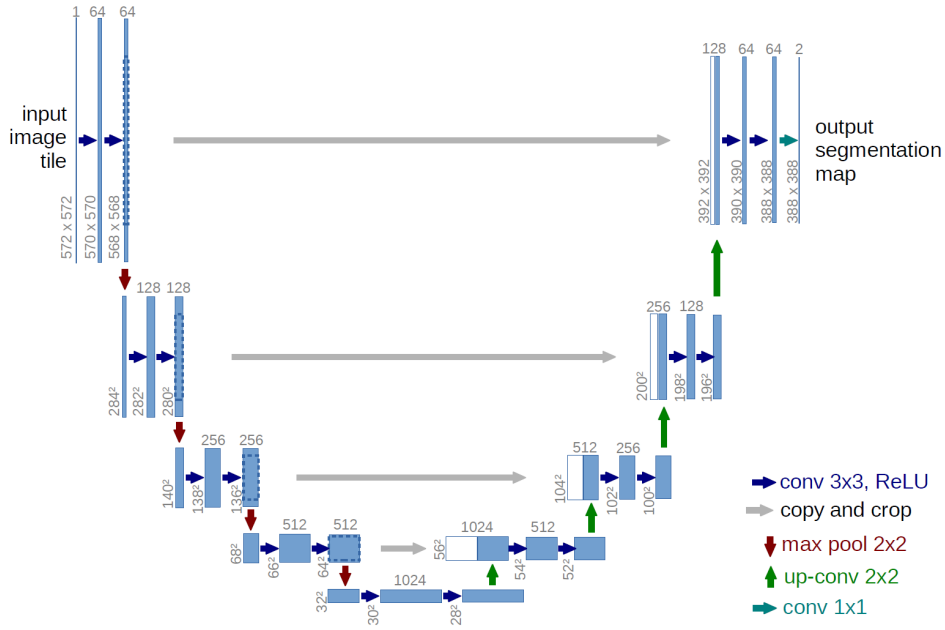

Figure 5: The U-Net architecture from Ronneberger et al [1]

Mean Average Precision, defined as:

$$mAP = \frac{1}{11} \sum_{r \in \{0.5, 0.55, \ldots, 1.0\}} AP_r \tag{4}$$

where the Average Precision at threshold r and Intersection Over Union (IOU) are defined:

$$AP_r = \frac{TruePositives}{(TruePositives + FalsePositives)}, \quad IOU(TruePositives) \geq r \tag{5}$$

$$IOU = \frac{Area\,of\,overlap}{Area\,of\,union} \tag{6}$$

We also used the pixel-wise F1 score defined as:

$$F_1\,score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

$$Precision = \frac{TruePositives}{(TruePositives + FalsePositives)} \tag{8}$$

$$Recall = \frac{TruePositives}{(TruePositives + FalseNegatives)} \tag{9}$$

5 Results

5.1 Segmentation Masks

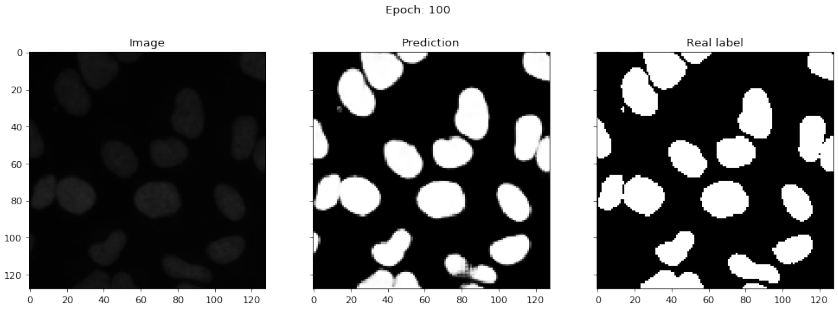

Figure 6: Segmentation mask for Kaggle dataset after 100 epochs

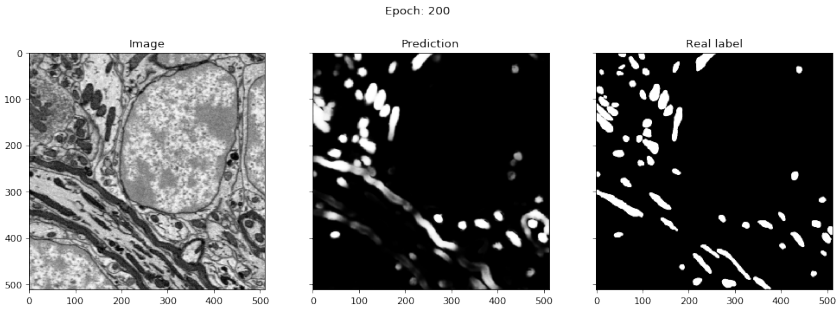

Figure 7: Segmentation mask for NCMIR dataset after 200 epochs

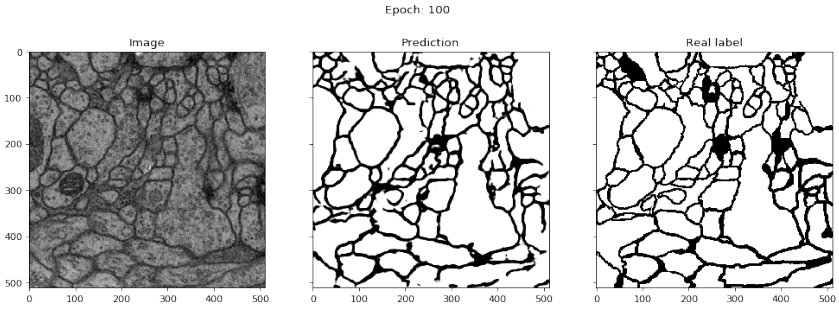

Figure 8: Segmentation mask for ISBI 2012 dataset after 100 epochs

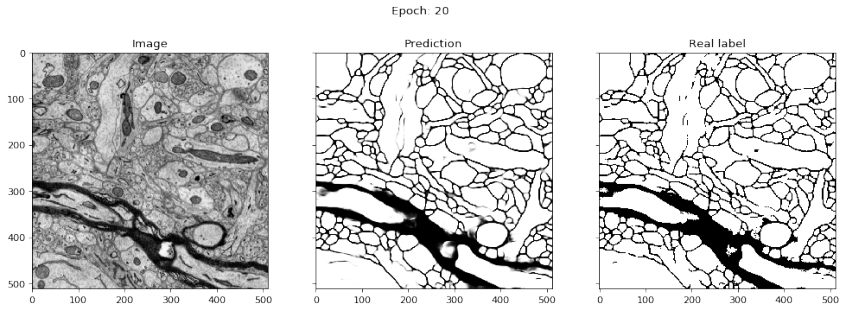

Figure 9: Segmentation mask for ISBI 2013 dataset after 20 epochs

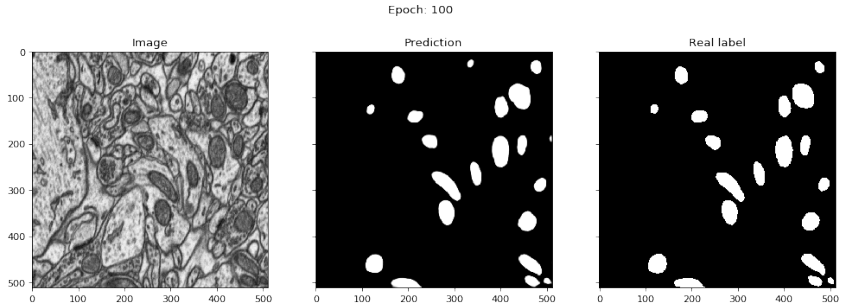

Figure 10: Segmentation mask for EPFL dataset after 100 epochs

5.2 Accuracy

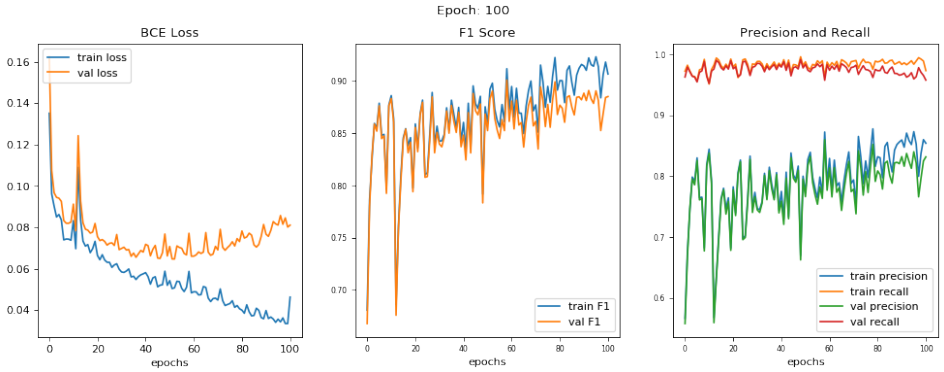

Figure 11: BCE Loss, F1 Score and Precision/Recall plots for Kaggle dataset

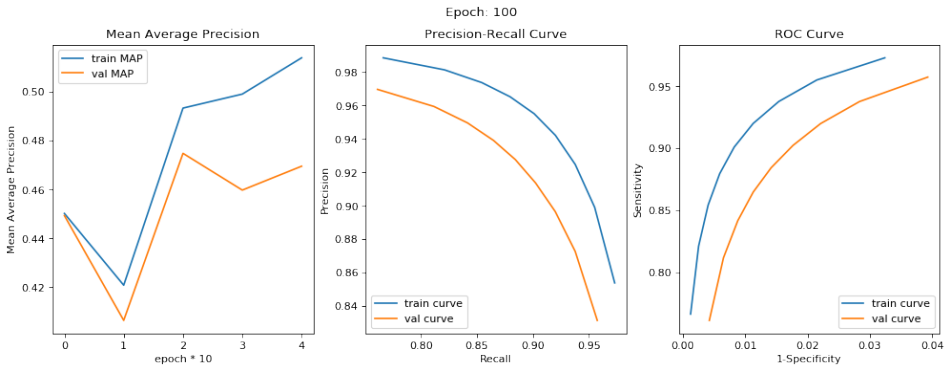

Figure 12: mAP, Precision/Recall curve and ROC curve for Kaggle dataset

Table 1: Evaluation results on validation sets after 100 training epochs

	Kaggle	ISBI 2012	ISBI 2013	NCMIR	EPFL
mAP	0.47	0.53	0.42	0.23	0.57
F1	0.89	0.93	0.97	0.63	0.95

5.3 Results and Model Discussion

The baseline results that we got were good even when we didn't include data augmentation techniques to boost the training dataset. The reasoning for this is likely because the images that we used were often directly related - taken with the same machine (Electron Microscope) and in fact cross sections of the same cell. Furthermore, the images were very high resolution, which meant that our network was able to get a lot out of each training image.

However, the values from the mAP (mean average precision) metric are noticeably lower than the other metric (F1 score). The reasoning for this is that the mean average precision is a lot more sensitive to individual classification errors. The F1 score is generally a better indicator of how well we did in the grand scheme of things, since it weights both precision and recall in its formula (see section 4.4 for formulas), while the mean average precision is a strong indicator of how well we do when classifying small details, getting significantly worse with each false positive that our model makes. Taking these two scores into account, it can be considered that our model is very good at locating where the masks should be, but makes a lot of errors on the edges compared to the target mask. Looking at the visual results that we've included in [section 5.1], just from inspection it is very clear that the general locations of every component is correct, and there are almost no blatant errors in the predictions in our model. However, especially in the datasets [7] and 8, the density and edges of the prediction mask are clearly different from the target. Although we only trained for 200 epochs (took about 2 hours on a 2GB GPU), which is nowhere near the amount of time that Ronneberger [1] recorded (10 hours on Nvidia 6GB GPU).

A complete collection of plots and segmentation output figures can be found in the *figures* directory included the code repository for this project.

5.4 Challenges

Analyzing this further, it is unclear how much of a problem unclear edges will pose in real applications in the biomedical field. It is possible that hand traced and stained images are already considered to be rough around the edges, and that the false positives and negatives that our model predicts are actually more accurate than the proposed target masks. It is also possible that this is a major shortcoming in applications that require very high precision, such as machine aided surgeries or diagnoses. As the model stands now, applications that focus on broad characteristics from the

images, such as studying cell movements in a culture after being exposed to different stimuli may be reliably labeled, but further work is required to extend into the larger biomedical field.

Some unexpected outcomes that we experienced were that validation and test accuracy actually decreased somewhat when including a significant amount of augmented data. We were not able to reach a conclusive decision on how to determine the optimal amount of augmented data to add to the dataset, as well as how drastic the augmentation can get before the results degrade significantly. We suspect that the reason we saw a decrease in accuracy after applying the augmented dataset was because the images were too similar to each other, and so the model became too specialized to our dataset (the validation images were too similar to the training images). In situations where we have a more diverse set of images or a larger dataset, the benefits of data augmentation in terms of versatility of classification may be more apparent. However, due to both time constraints and the limited amount of datasets that were available, we were not able to explore this adequately.

Unfortunately this specific model will need to be retrained to an extent for different tasks as well as different styled images, and we did not have the resources to improve our model to account for multiple features in its predictions and training. Some of the previous Kaggle winners [7] modified their architecture to account for borders, centers, and components in a holistic manner. Due to time restrictions, we ended up needing a new network to make predictions for each of those tasks separately.

5.5 Post-processing Discussion

Finally, our post processing attempt showed some promising results for a proof of concept implementation. The main problem seemed to be finding a reliable method of detecting how many clusters there were in the final mask. If we counted beforehand and used that number to seed the K-means algorithm, then there were very few errors on inspection. Of course, since our datasets did not include properly differentiated components on ambiguous structures, there was no way to determine a score of how well we did. We used the elbow method to approximate the number of clusters for our K-means algorithm, but the approximation in that method meant that error would always appear in our results [13]. (The colored patches seen in [section 3] do not differentiate touching structures as separate)

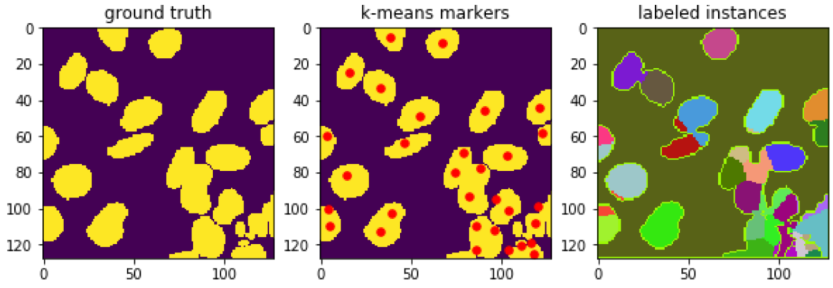

Figure 13: Initial Colormask Segmentation Attempt using K-means and Watershed

6 Future Work

It can be concluded that the U-net designed by Ronneberger [1] is very effective at processing contextual data and generalizing it through the features of similar images. Our model had problems with perfecting edges, but this may be improved with a longer training period. However, there are certain limitations in that the network cannot be easily generalizable across multiple different datasets at the same time; the system does not perform well to images that are too unfamiliar, and it is not easy to gauge the tolerances for data augmentation for different datasets. Post-processing techniques can to an extent distinguish components on generated masks, but the network itself does not have the capability to. These are areas that can be further explored in future work.

Additional work can also be done to include feature hierarchy into our network [7], which is what one of the contestants of the Kaggle competition used to further improve on results. The theory is that a hierarchy of features is more useful in ambiguous situations such as edges and touching borders in our case. It could significantly improve our mAP scores by compensating for our poor edge precision.

References

[1] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." In International Conference on Medical image computing and computer-assisted intervention, pp. 234-241. Springer, Cham, 2015.

[2] Çiçek, Özgün, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. "3D U-Net: learning dense volumetric segmentation from sparse annotation." In International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 424-432. Springer, Cham, 2016.

[3] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik "Rich feature hierarchies for accurate object detection and semantic segmentation" 2014

[4] Sadanandan SK, Ranefall P, Le Guyader S, Whlby C. Automated Training of Deep Convolutional Neural Networks for Cell Segmentation. Scientific Reports. 2017;7:7860. doi:10.1038/s41598-017-07599-6.

[5] Schlegl, Thomas, et al. "Fully Automated Segmentation of Hyperreflective Foci in Optical Coherence Tomography Images." arXiv preprint arXiv:1805.03278 (2018).

[6] Johnson, Jeremiah W. "Adapting Mask-RCNN for Automatic Nucleus Segmentation." arXiv preprint arXiv:1805.00500 (2018).

[7] Kaggle Data Science Bowl 2018 Dataset. Web link

[8] IEEE International Symposium on Biomedical Imaging Challenge 2012 Dataset. Web link

[9] IEEE International Symposium on Biomedical Imaging Challenge 2013 Dataset. Web link

[10] National Center for Microscopy and Imaging Research: Mitochondria dataset provided by Matthias Haberl. Web link

[11] École Polytechnique Fédérale de Lausanne Computer Vision Laboratory: Electron Microscopy Dataset. Web link