Fall 2020 Machine Learning and Imaging

Zebrafish Scale Detection and Illumination Optimization with Retina Net

Christian Richardson
11-24-2020

Abstract:

Zebrafish scale segmentation is a problem that has yet to be solved by known image processing approaches. The open source image segmentation platform Ilastik[1] was previously used to generate masks of the zebrafish scale, but an approach that can identify individual scales in dense environments is desired. Zebrafish scales are relatively thick three-dimensional structures that must be imaged in 2D slices which are compiled into 3-dimensional tiff stacks. Additionally, there is variation in illumination across the scale due to the three-dimensional structure of the specimen, which leads to imbalance in illumination particularly towards the posterior end of the scale. Segmenting images of these scales is difficult due to the low contrast of the images in areas with poor illumination, and the close proximity that scales share, which generally leads to multiple scales being merged into a single connected component when image segmentation is used. I propose an object detection-based solution, where the object detection model Retina Net is used to regress bounding boxes around individual scales. Retina Net is modified to include a multiply layer that element-wise multiples the sample intensity map, or the image of a scale, with a learned illumination map. This process models incoherent illumination, where the element-wise multiplication of a transparency and an illumination map results in the acquired image, which is later passed through a point spread function. Unfortunately, I was not able to implement a functional Retina Net model that includes a custom illumination layer. Instead, the classic Retina Net model was used to perform object detection on Zebrafish scale images.

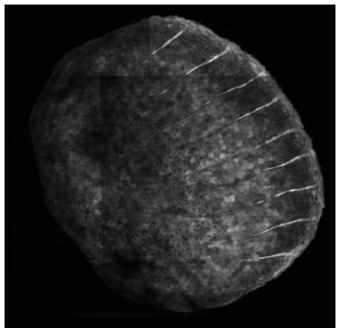

Figure 1 Max-Projection of Cleaned Zebrafish Scale Image

Introduction:

How Zebrafish scales regenerate after plucking is a question that is being investigated by the Di Talia lab at Duke. It is believed that waves of activity of the protein ERK are responsible for coordinating the growth of regenerating scales. To acquire data from individual nuclei corresponding to a particular scale of interest (i.e. the regenerating scale), image segmentation must be employed to isolate nuclei corresponding to the scale of interest. However, image segmentation platforms such as Ilastik[1], which uses a random forest-based algorithm along with various features extracted from images on different scales, are problematic because scales share close proximity and are often merged together during the process of image segmentation. This results in connected components that span multiple scales and lead to acquisition of data irrelevant to the scale of interest. The problem is further complicated by the need for 3D segmentation, since the Zebrafish scale images are actually 3D stacks of images that each contain information from portions of scales. The scale of interest needs to be 'tracked' across slices throughout the image stack to acquire all of the data pertaining to the scale of interest. Any merging of the scale of interest's connected component mask with connected component masks belonging to other scales (Fig. 1) greatly complicates tracking of the scale of interest, making image segmentation a potentially ineffective approach to solving this problem.

Figure 2 Proximity of Scales makes Image Segmentation Difficult

Due to the challenges faced with zebrafish scale segmentation using image segmentation-based approaches, I decided to attempt to use object detection to identify and later track zebrafish scales across Z-slices. Object detection has experienced a renaissance in recent years due to advancements in deep learning and the technology required for deep learning. Object detection frameworks can be categorized into one-step and two-step detectors, based on whether a candidate object generation step is included in the algorithm. Two-step detectors, such as Faster-RCNN, generate candidate object locations before moving on to bounding box regression and classification. One stage detectors, including YOLO and Retina Net, skip candidate object generation and instead sample from a dense grid of anchor boxes, applying classification and bounding box regression directly to the input image[2]. Retina Net in particular is interesting because it uses a novel loss function, called focal loss, that emphasizes "harder" training examples and diverts efforts away from easy-to-classify examples, which comprise most of the samples encountered during object detection training[2]. Retina Net was the first one stage detector to produce comparable results to state-of-the-art two stage detectors. Because Retina Net is a lightweight, fast framework that produces high accuracy results, I decided to use Retina Net for detection of Zebrafish scales.

$$Focal\ Loss(p_t) = -a_t(1-p_t)^{\gamma}\log(p_t)$$

For the physical layer component of this project, I sought to model incoherent illumination, specifically the multiplication between an illumination matrix and a reflectivity or absorption map which in our case is the images of the Zebrafish scale. The Zebrafish scale images suffer from poor contrast due to low illumination levels, which vary across the XY plane of the image. The illumination matrix would be learned along with the rest of the CNN, providing an optimal illumination for object detection. Unfortunately, I was not able to figure out how to implement the physical layer in a model that also takes as inputs the zebrafish scale images. However, I was able to perform object detection on the fish scale images using an unmodified Retina Net implementation. Ultimately, I believe that if I were able to implement the physical layer the optimal illumination map would improve the performance of Retina Net on the Zebrafish scale images.

Related Work:

The use of a physical layer for modeling object illumination and detection as part of a convolutional neural network was pioneered by Dr. Horstmeyer in the paper *Convolutional Neural Networks that Teach Microscopes How to Image*[3]. In this work, a full incoherent imaging process is simulated through the use of a point spread function convolution that is applied after the sample illumination is performed using an element-wise matrix multiplication, identical to the element-wise matrix multiplication I attempted to create. Thus, my approach is essentially a simplified version of the physical layer implemented in Dr. Horstmeyer's paper. Other works involving optimization of illumination have been made, including optimization of an illumination pattern for single-shot fourier ptychographic microscopy[5]. Another work related to illumination is indoor lighting prediction[4], which is in some ways the inverse of what I attempted to do, where the ground-truth illumination is predicted as the output of the model, in contrast to my approach where illumination is modulated to achieve an optimal output. These works all use convolutional neural networks and automatic differentiation to optimize illumination of a sample at some stage of the image acquisition process and are similar in this regard to my work presented here.

Methods:

To create annotations for Retina Net-based bounding box regression, I first had to process the Zebrafish scale mask images I received to obtain object bounding boxes. Additionally, images were resized to dimensions of 1000,1000, because each image needed to be the same size for the element-wise multiplication of an illumination matrix with the input image that occurs later. To obtain bounding boxes from the mask images I received, the python module opencv was used. Opencv commands *contour* and *boundingRect* were used to obtain each connected component in a mask image, then estimate the corners of a bounding box around the connected components. These coordinates were then automatically added to a csv file that contains the required information for training Retina Net.

It should be mentioned that originally I planned to use the EfficientDet architecture for object detection, but due to issues with getting the keras implementation of EfficientDet to work, I switched to RetinaNet. Two RetinaNet implementations were used throughout this project. One, obtained from the github user fizyr[7], was used first and is the functional model that I use to produce the resulting bounded Zebrafish scale. The second implementation[6] comes from the keras website and was designed to be used with the COCO2017 dataset. While both implementations work fine without any modification to the source code, I was unable to configure either model to work with my data while also handling the task of learning an optimal illumination pattern. A detailed explanation of why I was unable to implement the physical layer is given below.

Results:

For this project, I had two primary goals: to implement a physical layer modeling the multiplication of an object with an illumination matrix, and to accurately regress bounding boxes around zebrafish scales in images. The first goal, to implement a physical layer, was not successful due to numerous issues encountered with modifying Retina Net implementations to include this physical layer. The first implementation I worked with, provided by the github user fizyr[7], worked for performing object detection on my data set, but was intractable as far as modifying the implementation at the level required to implement a physical layer. I intended to implement the physical layer before the input image is processed by the backbone network of Retina Net, in what would be the first stage of the new network. However, I had great difficulty figuring out how images are opened and fed into the backbone network with this implementation. I ultimately tried making a new backbone that included the physical layer, but I ran into bugs trying to load the new backbone model within Retina Net.

Because of the issues I encountered with the first implementation, I instead tried to use another Retina Net implementation, courtesy of the Keras website. This implementation was far more transparent and I was able to add a multiply layer (which comprises the physical layer I was aiming for), however I was not able to figure out how to load, preprocess, and input my images and annotations into the network. This Retina Net implementation used data that was provided in the form of a tensorflow data set, and was amiable to the number of tensorflow preprocessing operations applied to the data, which I was not able to decipher in time. I do believe I was close to figuring out how to use this implementation with a physical layer, but towards the end of my debugging the code began to take a very long time to run on the step preceding training, and I was not able to figure out how to speed it up or correct whatever error caused the model to have very long run times.

The second goal I had for this project was to accurately predict bounding boxes for zebrafish scales. The hope was that I could get highly accurate, 'tight' bounding boxes that enclose the entirety of a scale while leaving minimal extra room belonging to background. This objective was not achieved entirely, bounding box regression was performed on the zebrafish scale images but the boxes did not cover all portions of the scale and often covered large amounts of background. The training images used were not ideal for the goal of full zebrafish scale segmentation because the training images are 'clean' images, or images where only the scale of interest is present. The idea was to use object detection for cleaning images, where there would be several scales present within an image at a time. Because the training data was only comprised of cleaned images, the object detection task is simplified, yet the bounding boxes regressed are still not ideal. It is possible that the training bounding boxes used to train the network were of poor quality due to inaccuracies in the masked images provided and the automated nature by which bounding boxes were generated.

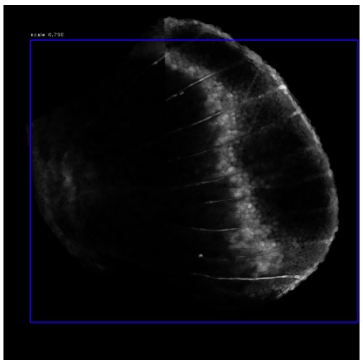

Figure 3 Zebrafish Scale Bounding Box Prediction

Training was performed in 2 epochs, with 100 steps per epoch and a batch size of 32. The regression accuracy for bounding box regression stayed between 40% and 30% for most of the training process. A mAP score of .502 was obtained for the training process. The low bounding box regression accuracy implies that the network had a hard time with determining bounding boxes for the scales. Additionally, regression accuracy began to decline at a certain point in the training process, meaning it is possible that the Retina Net architecture was not able to provide better bounding boxes for the images provided. The low number of total steps was chosen because of time constraints and to avoid overfitting. I am not sure if it would have been better to train the network for longer, or if it reached its maximum performance given the data.

Figure 4 Batch Regression Accuracy

Discussion:

In summary, the results of this project were not what I hoped for, in terms of implementing the physical layer and getting reasonable bounding box predictions from Retina Net. Issues with bugs and difficulties with adapting existing implementations of Retina Net bogged down my progress. I did not have enough time to optimize the training of my model due to time spent trying to implement the physical layer. I believe that with more time, I would have been able to implement a physical illumination layer and that this layer would have improved performance of object detection. This project was a learning experience for me, and if I could start over from the beginning, I would probably have asked for more assistance. While I was not able to implement my physical layer or get high-accuracy bounding box prediction for the zebrafish scale images, I appreciate the experience of learning about how more complicated architectures like Retina Net are implemented, and the challenges faced in an object detection project. This project highlights the difficulty inherent in modifying implementations of complex networks.

References:

1. *ilastik: interactive machine learning for (bio)image analysis*
   Stuart Berg, Dominik Kutra, Thorben Kroeger, Christoph N. Straehle, Bernhard X. Kausler, Carsten Haubold, Martin Schiegg, Janez Ales, Thorsten Beier, Markus Rudy, Kemal Eren, Jaime I Cervantes, Buote Xu, Fynn Beuttenmueller, Adrian Wolny, Chong Zhang, Ullrich Koethe, Fred A. Hamprecht & Anna Kreshuk
   in: Nature Methods, (2019) Link at publisher, BibTex file
2. Focal Loss for Dense Object Detection
3. Convolutional neural networks that teach microscopes how to image
4. Neural Illumination: Lighting Prediction for Indoor Environments
5. Illumination Pattern Design with Deep Learning for Single-Shot Fourier Ptychographic Microscopy
6. https://keras.io/examples/vision/retinanet/
7. https://github.com/fizyr/keras-retinanet