January 2022

Computational Single-Cell Analysis Of Confocal Fluorescence Images With Dapi-Generated Masks

Munirah Alduhailan
Wayne State University

COMPUTATIONAL SINGLE-CELL ANALYSIS OF CONFOCAL FLUORESCENCE IMAGES WITH DAPI-GENERATED MASKS

by

MUNIRAH AL-DUHAILAN

THESIS

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfilment of the requirements

for the degree of

MASTER OF SCIENCE

2022

MAJOR: PHYSICS

Approved By:

_____

Advisor Date

ACKNOWLEDGEMENTS

First, I would like to thank Prof Christopher V Kelly, who guided me through these projects. He provided me with invaluable advice and helped me during difficult periods. His motivation and assistance contributed tremendously to the successful completion of the project. Furthermore, I would like to thank my family and friends for their support, without which I could not have succeeded in completing this project. I would also like to thank the Saudi Arabian Cultural Mission (SACM) for their financial support. Last but not least, I thank everyone who helped and motivated me through this project.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

Digital image processing techniques allow for the extraction of quantifiable information from complex images. Image analysis tasks range from reading simple bar-coded tags [1] to more complex facial recognition [2]. Scientific image analysis is important because it enhances quantitative information for hypothesis testing. Moreover, fluorescence imaging provides molecular specificity and single-molecule resolution. Improved image analysis of fluorescence images can, thus, enable the assessment of single-cell behaviors and metabolic pathways.

Image processing methods are rapidly advancing with the application of artificial neural network algorithms that can recognize complex patterns and analyze information by grouping and categorizing raw input with machine perception. Neural networks recognize numerical patterns encoded in vectors; therefore, all input data must be converted into vectors so that their patterns can be recognized.

Segmentation of complex biological images, in which regions of the image are distinctly labeled, is the first step in image analysis. Ilastik is a software designed to perform image segmentation using neural networks [3] [4] [5]. Ilastik enables the classification and segmentation of images in a consolidated and easy fashion, as it does not require any expertise in image processing or neural network programming. Ilastik handles complex textures by employing an intuitive graphical user interface.

The results from Ilastik can be analyzed using general purpose programming languages for the custom needs of each experiment. Python is an exemplar programing language for this purpose with numerous scientific applications, good programing flexibility, ease of programming, and efficient performance.

This thesis reports on the development of a workflow that incorporates the Ilastik neural network software to analyze fluorescence images for the construction of single-cell data. We combined multiple colored fluorescence tissue images to analyze the growth and metabolism of lipid droplets (LDs), which are spherical organelles comprising a neutral lipid core surrounded by proteins and a phospholipid monolayer. They play a fundamental role in the regulation of cellular homeostasis. LDs

grow and detach from the endoplasmic reticulum during energy storage. The LDs reduce in size upon times of energy need via the lipolysis of triglycerides and free fatty acid release. Many prokaryotic and eukaryotic cells produce LDs to regulate their metabolism.

The mobilization of triglycerides stored within LDs is regulated by the protein, α/β-hydrolase domain-containing protein 5 (ABHD5), which activates an enzyme adipose triglyceride lipase (ATGL). ATGL is the rate-limiting enzyme involved in the breakdown of triglycerides from LDs into free fatty acids. Mutations in ABHD5 are associated with lipodystrophy, lipids accumulating in cells, and Chanarin–Dorfman syndrome.

ABHD5's ability to stimulate ATGL is affected by various natural and synthetic pathways. Isoproterenol (ISO), for example, is a positive control that stimulates lipolysis through an androgen receptor agonist. Compound SR3420 was identified using a high-throughput screen that disrupted the interaction between ABHD5 and PLINs and was later shown to stimulate the lipolysis of brown adipocytes and muscles [6]. SR3420 is typically delivered to cells via the delivery agent, dimethyl sulfoxide (DMSO). DMSO is one of the most used solvents in cell biology, and it has a high polarity and can dissolve ionic and non-ionic compounds [7] [8].

Finally, forskolin (FSK) is a lipid-soluble compound that can penetrate cell membranes and stimulate the enzyme, adenylate cyclase, which increases intracellular cyclic adenosine monophosphate levels to stimulate lipolysis [9]. This study reports the use of neural-network-based fluorescence image analysis to measure single-cell triglyceride storage and ABHD5 expression upon stimulation with ISO, SR3420, DMSO, and FSK.

Chapter 2

Methods

A workflow for the analysis of two-dimensional (2D) fluorescence images with single-cell masks is shown Fig. 2.1. The images to be analyzed included adherent COS7 cells transfected to express ABHD5-mCherry. The night before imaging, the cells were exposed to oleic acid and incorporated into the LDs. Three hours before imaging, the cells were incubated with the lipophilic fluorophore, BODIPY, and one of the four stimuli. Lipolysis (i.e., BODIPY incorporation) and ABHD5 expression levels were stimulated to varying degrees. The cells were imaged by using a spinning disk confocal microscope (Olympus IX81 DSU) and an sCMOS camera (Andor Zyla 4.2). The images were saved as 8-bit TIF images and were input into the imaging pipeline developed for this study. The analysis process began with Ilastik, owing to its built-in capabilities for image analysis via neural network incorporation. We identified individual pixel classes and sorted them into multi-pixel objects. The Ilastik objects were then loaded into Python with the H5 file type. The center-of-mass was calculated from each DAPI object and used to make single-cell masks using a Voronoi diagram. Finally, the fluorescence images in the non-DAPI channels were analyzed with single-cell masks to reveal the single-cell fluorescence brightness and variability.

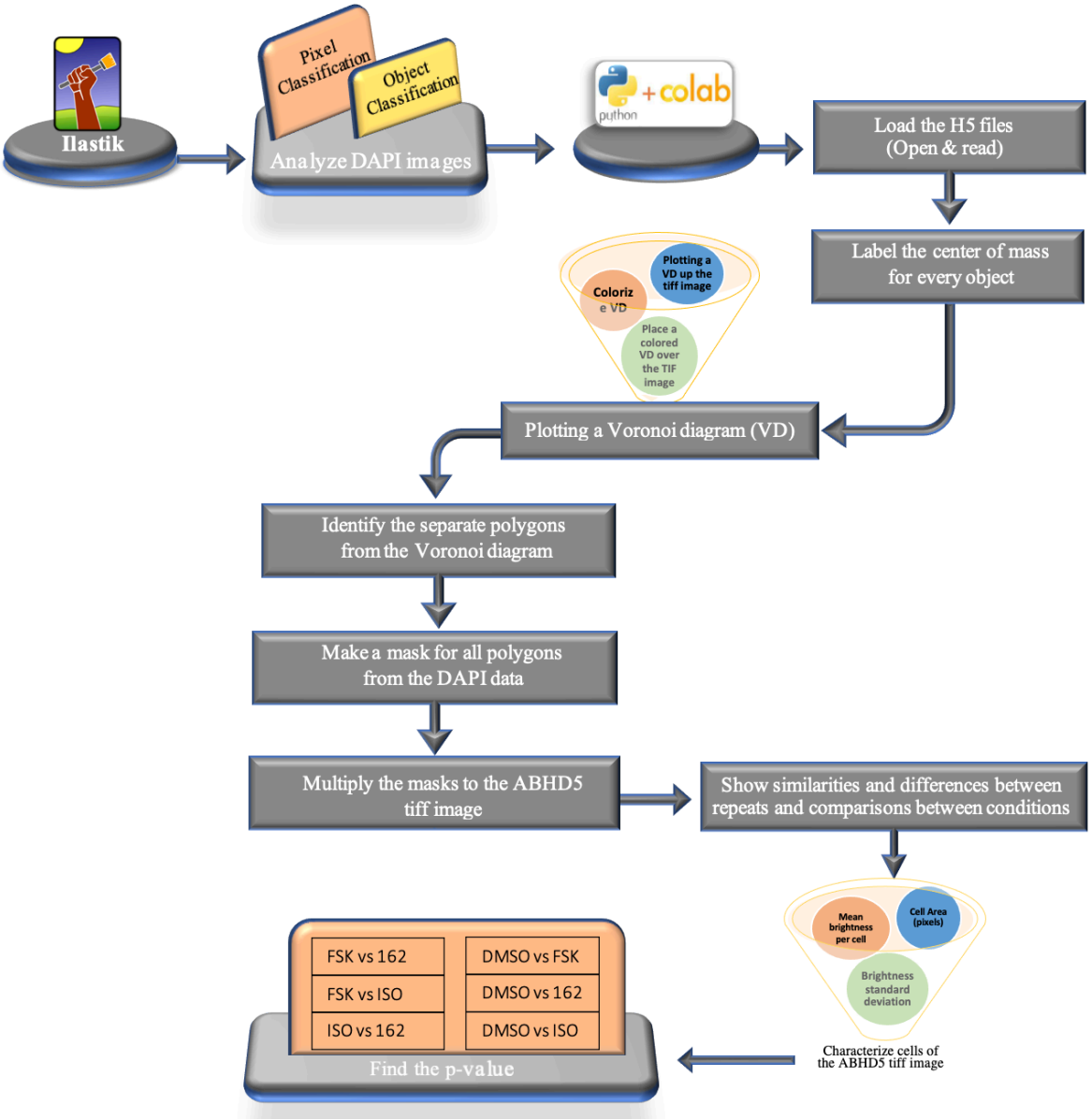

Figure 2.1: Two-dimensional analysis of images in Ilastik and Python. We used Ilastik because of its built-in capabilities for image analysis with neural networks. We performed pixel classification and object classification of the DAPI images to identify single cells as separate objects. The DAPI objects and raw data from other colors were imported into Python via H5 and TIF files, respectively. The centers-of-mass for each DAPI object were used to create Voronoi diagrams and isolate image regions corresponding to each cell. Next, the other color channels were analyzed using the Voronoi diagram mask. This process was repeated with images from all repetitions and conditions to test the effects of stimulating conditions with statistical hypothesis testing.

2.1. Ilastik pixel classification

Pixel classification was performed using Ilastik with fluorescence DAPI images by analyzing each pixel and its neighbors' fluorescence intensities. Pixels corresponding to each class were visually identified and graphically labeled, including classes used to represent the cell nucleus, cytoplasm, and extra-cellular spaces. These user-defined classes were used to train the neural network, which was then applied to all pixels across all DAPI images to perform the pixel classification, as shown in Fig. 2.2.

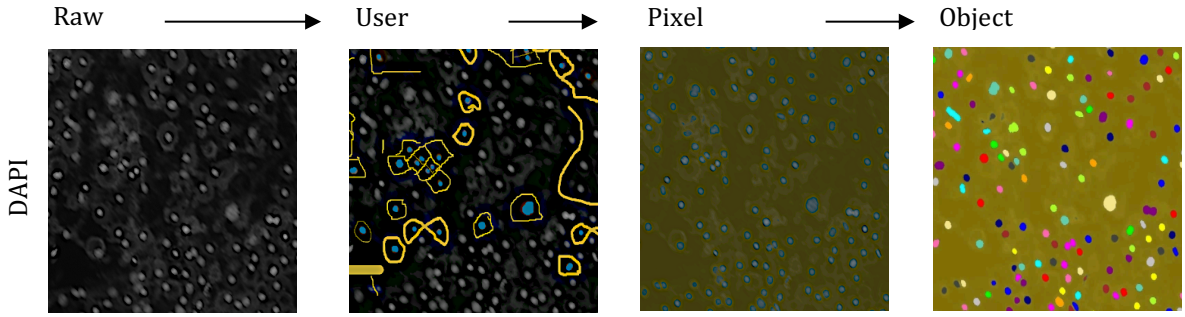

Figure 2.2: Ilastik analysis of DAPI images was used to determine individual cells through pixel classification and object classification. The process started with the raw image, with which we used a computer mouse to draw the shapes as annotations on the image. These annotations were used to train the neural network for pixel classification across all DAPI images. The results from pixel classification were further analyzed to perform object classification for the identification of single cells; therefore, Ilastik recognized and saved the individual, separate objects.

Pixel classification workflow consisted of five major steps (Figure 3):

- Create a new project by selecting "Pixel Classification" on the graphical user interface, after which the system prompts for a filename to save the project.
- Load the data into the project.
    - Load raw 8-bit data (2D) by adding separate images.
    - In the dataset properties editor, the correct order of the X and Y axes should be entered as necessary in the "Raw data" field.
- Select the pixel features and their scales to distinguish between different classes for the neural network using "Feature Selection."
- Select annotations from the image to train a classifier and label each pixel class to be separated (e.g., background, dim, medium, or bright).

- Export the results as an H5 pixel classification using “Prediction Export,” then, using “Export Setting,” choose “Probabilities” in the source.

The pixel classification process performs semantic segmentation rather than instance segmentation and outputs a probability map for each class rather than for individual objects.

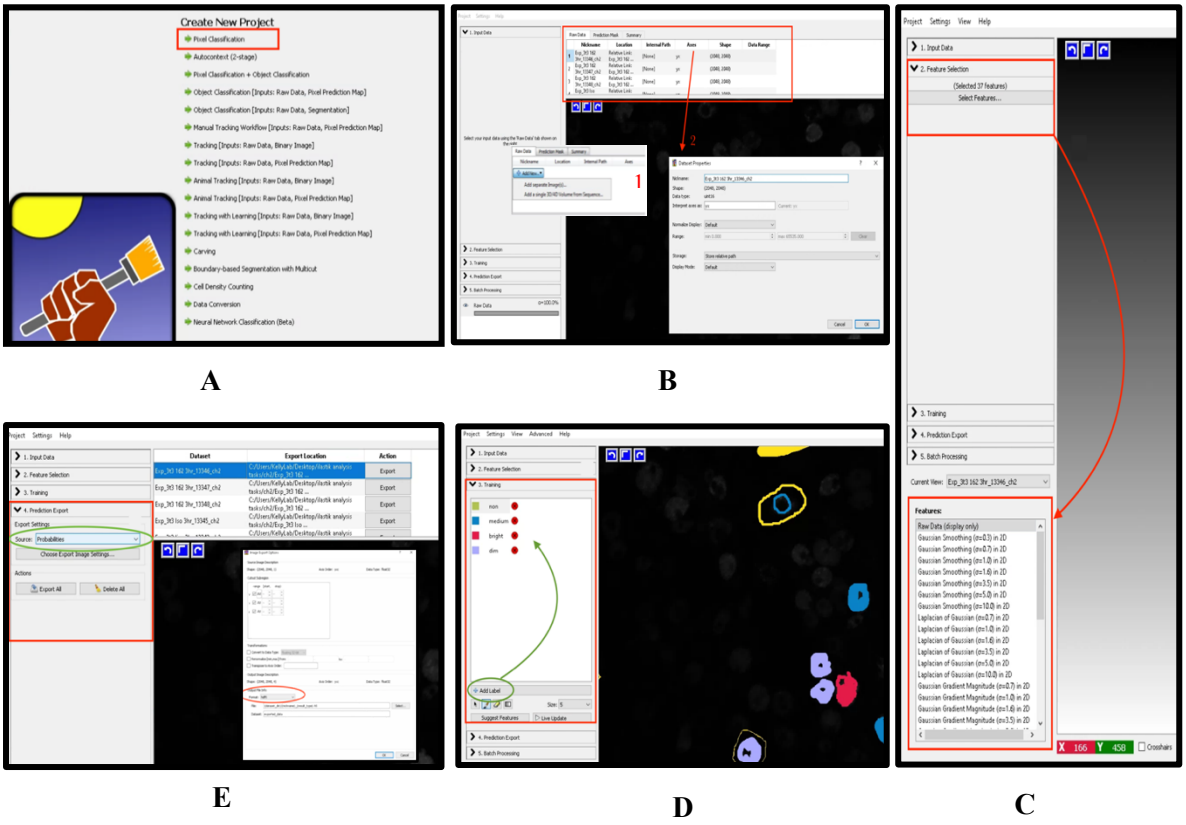

Figure 2.3: Pixel classification workflow. Pixels are categorized according to their features and user annotations: (A) graphical user interface used to create a new project for pixel classification; (B) loading raw data to be classified into a project and checking the order of axes; (C) recognizing different pixel classes; (D) training a classifier by manually drawing annotations; and (E) exporting the results of this workflow (pixel classification) as H5 files.

2.2. Ilastik object classification

After individual pixels were classified, we identified the multi-pixel objects representing each nucleus and converted the probability map into individual objects to identify individual cells from the DAPI images, as shown in Fig. 2.2.

The object classification workflow comprised five major steps, as shown in Fig. 2.4:

- Create a new project by choosing "Object Classification [Inputs: Raw Data, Pixel Prediction Map]," by which the user is prompted to name the new file to be saved in the project.

- Load 2D raw data and the pre-computed probability map using “Pixel Classification,” and in "Threshold and size filter,"
    - Select the "Simple" filter method,
    - Select the input channel corresponding to the object,
    - Adjust the threshold value and the filter scale, after which the resulting connected components of the foreground pixels are assigned random colors.

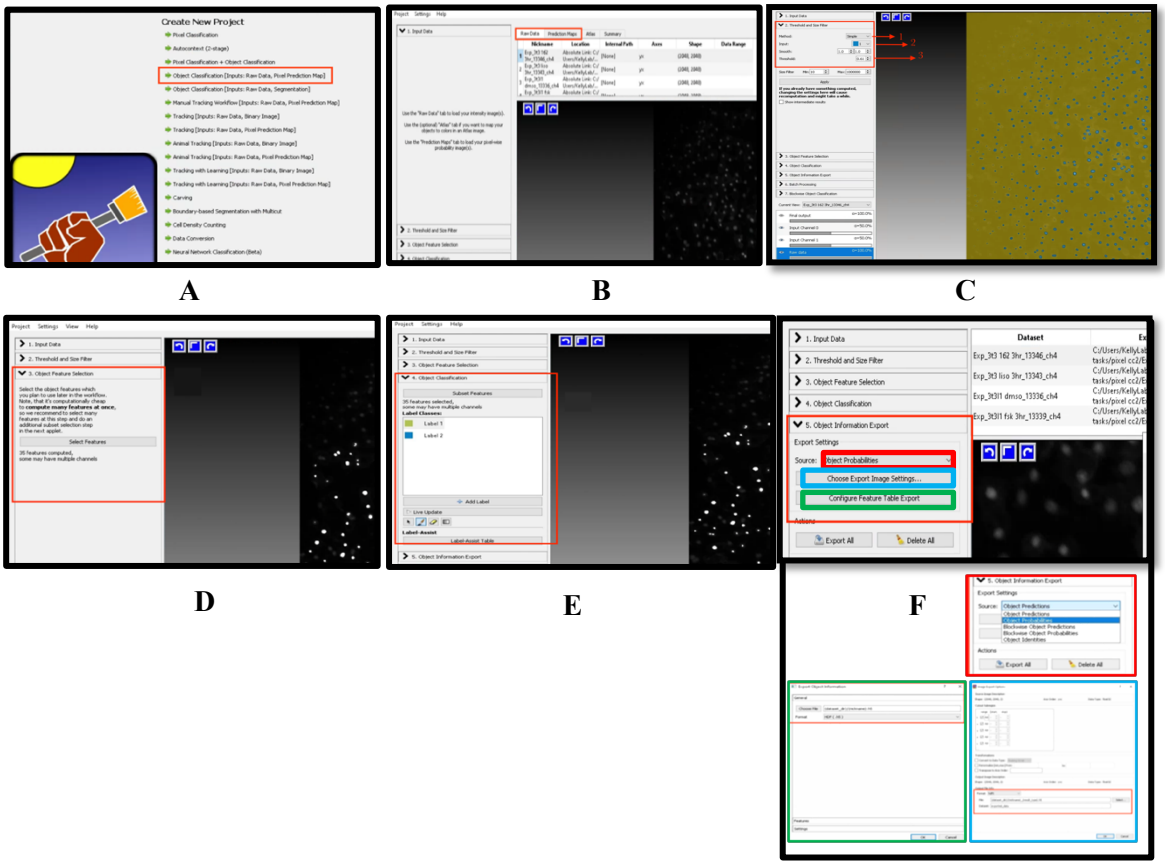

Figure 2.4: Object classification workflow to identify individual cells and transform the probability map into individual objects based on the object-level features and user annotations: (A) graphical user interface used to create a new classification project; (B) loading raw data and a pre-computed probability map of pixel classification into the project; (C) applying the selected threshold and size filter, and the resulting connected components of foreground pixels assigned random colors; (D) inferring the object type; (E) training a classifier by labeling a few cells for each class; and (F) exporting the results of the workflow (object classification) as one H5 file for each DAPI image.

- Select "All excl. Location" in the “Object Feature Selection” area to infer the object type.
- Label a few cells for each class in "Object classification" based on the features, as computed in the previous applet.
- Select “Object Probabilities” to the image export in the export source and export images and tables as H5 files in "Choose Export Image Settings "and "Configure Feature Table Export" to obtain a table that includes all information about the objects used during classification as HDF5.

2.3. Creating single-cell masks

The DAPI object classification from Ilastik was reduced to a single center-of-mass point for each object by importing the H5 files from Ilastik for each DAPI image into Python. The center of mass of each DAPI object became the input to a Voronoi diagram (i.e., Dirichlet tessellation) across the entire image. Each region from the Voronoi image (i.e., Dirichlet regions, Thiessen polytopes, or Voronoi polygons) corresponded to a region of the image for a single cell, as shown in Fig 2.5.

Each Voronoi polygon was converted to the same size as that of the TIFF image for easy multiplication and single-cell isolation. All single-cell masks were then saved in the "all_masks" array for application to other colored images, including those of BODIPY and ABHD5.

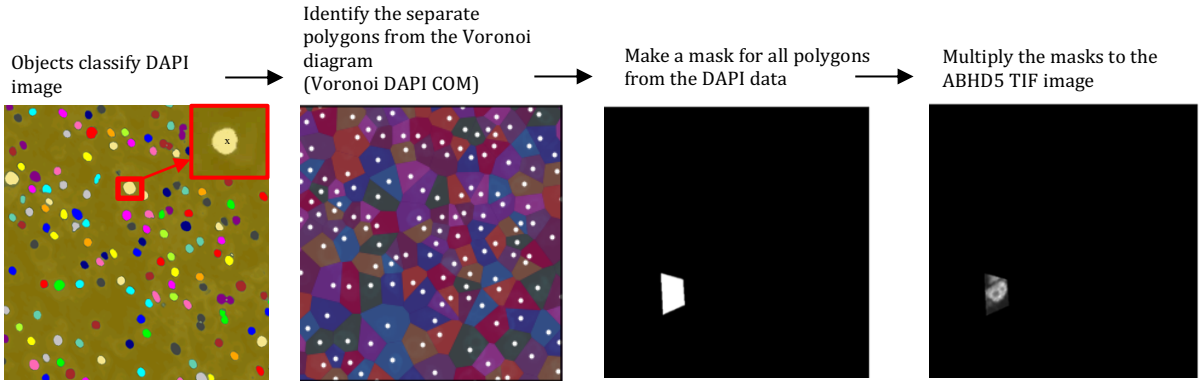

Figure 2.5: Analysis of additional color channels using single-cell Voronoi diagram mask. The Voronoi diagram was analyzed from the center-of-mass of each DAPI object, and we identified separate polygons and constructed a mask the same size as that of the TIFF image with ones for polygons and zeros otherwise. Upon TIFF-image multiplication and using a mask that resulted in zero for all areas not corresponding to the particular cell of interest, we isolated the fluorescence data under the mask.

2.4. Analysis of fluorescence images with single-cell masks

ABHD5 or BODIPY data were loaded, and masks were applied. The images were then analyzed under each mask to calculate the cell area, mean brightness, and brightness standard deviation of the ABHD5 and BODIPY per cell by adding code that finds the pixel inside the polygon. Furthermore, if there is a point inside the polygon, the code changes the values in the pixels to one. Therefore, it is possible to see under the mask. Histograms of data were created to characterize the distribution of single-cell properties, as shown in Fig. S1. Experimental analysis was then performed, and we determined whether the four different conditions resulted in significantly different cell brightness under conditions of three or four repetitions.

Additionally, swarm plots were created to show how the different conditions (i.e., DMSO, FSK, 162,and ISO) resulted in significantly different results, as shown in Fig. 2.6.

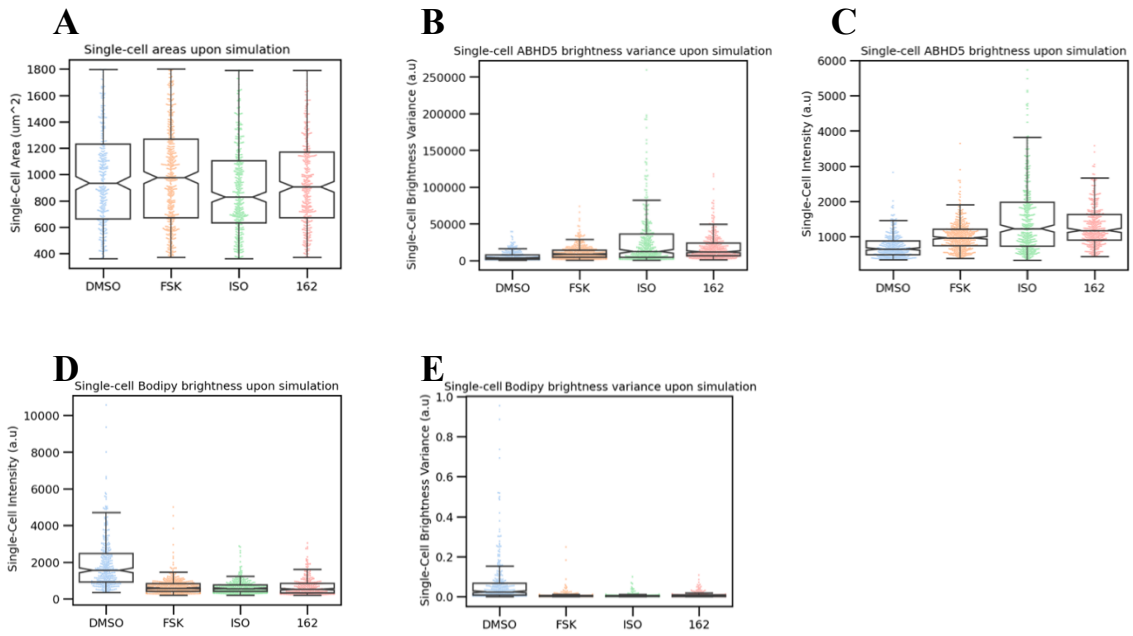

Figure 2.6: Analyses of single-cell brightness, brightness variance, and areas for each cell's ABHD5 and BODIPY fluorescence from DAPI-generated masks. Swarm plots show the single-cell and population differences for each condition (i.e., DMSO, FSK, 126, and ISO). The four conditions were compared, and all comparisons provided p-values less than 0.001. apart from the comparison of ISO and 162 (0.0017 in ABHD5), between ISO and FSK (0.18 in BODIPY), between FSK and 162 (0.88 in BODIPY), and between ISO and 162 (0.32 in BODIPY).

2.5. Plotting of results and statistical testing

Checking the confidence levels of the various conditions provided independent mean values of cell brightness (see Fig. 2.6). The p-values of six different sets of comparisons were thus calculated (i.e., FSK vs. ISO, FSK vs. 162, ISO vs. 162, DMSO vs. 162, DMSO vs. FSK, and DMSO vs. ISO) for both ABHD5 and BODIPY to determine whether the null hypothesis could be accepted or rejected. Additionally, a two-tailed test was applied using Python to identify any differences between the datasets.

We used p = 5% as the threshold for a good decision. If the p-value from the statistical analysis is lower than 0.05, there is a significant difference between datasets, and we can reject the null hypothesis. However, if the p-value is greater than 0.05, then there is no difference between the datasets. In such a case, we will likely not be able to reject the null hypothesis.

The t-test is used to analyze the average of the difference between the means of the samples and provides the difference between the two measures within a normal range. However, the p-value is performed to gain confidence that we can reject the indifference between the averages of the two samples. Thus, the p-value focuses on the extreme side of the sample to provide an extreme result.

Chapter 3

Results and discussion

Ilastik-supervised machine-learning software was used to perform pixel and object classification on DAPI images, resulting in recognized individual objects. A Voronoi diagram from the DAPI objects masked the entire image into regions corresponding to each cell and analyzed the color channels by changing the pixel values to one if there was a point inside the polygon. Moreover, the single-cell swarm plots captured the stimulation condition of ABHD5 expression, and the p-value results of all sets of comparisons were found to be less than 0.001, apart from ISO vs. 162 = 0.0017. This implies that there was no difference between ISO and 162. Finally, because our method could be easily extended to more images, we ingested hundreds more for the automated analysis of treatment concentrations.

A limitation of our study is that it depended largely on the user expertise of neural networks. For example, a user with slightly different training habits or familiarity with other DAPI analysis automation software and threshold identification would probably produce a slightly different result. Nevertheless, our findings contribute to a better understanding of neural-network-based fluorescence image analysis to measure single-cell triglyceride storage and ABHD5 expression upon stimulation with ISO, SR3420, DMSO, and FSK. Additionally, the results of the image segmentation produced by Ilastik were reliable. Furthermore, the Ilastik method described herein may allow the simplified identification of significant biological differences in ABHD5 stimulation in low-contrast images in a considerably less time than with conventional methods.

We formulated an algorithm that can be applied to large datasets automatically, which allowed us to test variable dosages and produce a dose-response script (e.g., obtaining a swarm plot with BODIPY vs. SR concentration and quantifying BODIPY changes). As a next step, we plan to consider other types of ABHD5 for analysis. It is possible that mutant ABHD5 may have a different response; hence, the efficacy of ABHD5 behaviors should be examined next.

Chapter 4

Conclusion

In this study, we analyzed multi-color fluorescence images of tissues to determine the growth and metabolism of lipid droplets. Our study makes a significant contribution to the literature, as our novel neural-network-based image analysis technique software, Ilastik, offers simple and detailed image processing of fluorescence images to molecular specificity and single-molecule resolution. Furthermore, Python scripting provides additional interpretations of the Ilastik results.

APPENDIX A

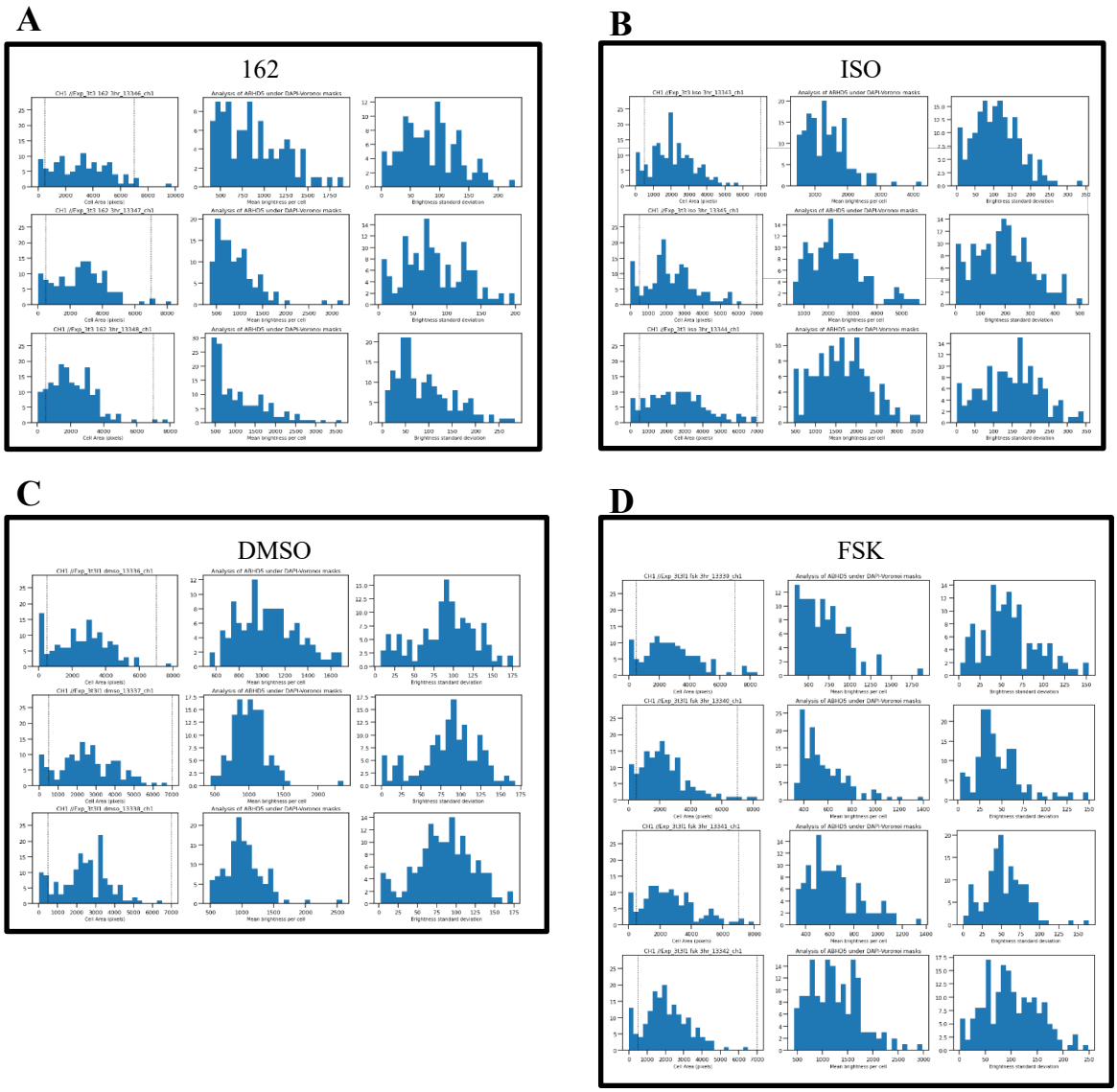

Figure S1: Analyses of cells stimulated by (A) 162, (B) ISO, (C) DMSO, and (D) FSK. We characterized the cells of the ABHD5 TIFF images by calculating cell area (pixels), mean brightness per cell, and brightness standard deviation to show similarities and differences between repeats (i.e., files with the same name but different numbers).

APPENDIX B

```
# -*- coding: utf-8 -*-
"""Munirah code ch4 + ch 1
Automatically generated by Colaboratory.
Original file is located at
    https://colab.research.google.com/drive/1z9eYc-
rbmtgeBRjRepkPBxx2MYDXPfTa
"""
from google.colab import drive
drive.mount('/content/drive')
#Import these libraries
import os
import h5py as hh
import matplotlib.pyplot as plt
import numpy as np
from scipy.spatial import Voronoi, voronoi_plot_2d
import pandas as pd
from scipy import ndimage
import matplotlib
import tifffile as tiff
from pathlib import Path
"""# **Loading the h5 files (open and read)**"""
target_dir = "/content/drive/MyDrive/CH4CH1 " #All ch1.tif,
ch4.tif, and H5 files in this folder
 #We made this a larger range to get more data analyzed.
analysis_range = [700,1300]
```

```
#listing the files in the target directory of channel 4
files_list = os.listdir(target_dir)
all_ch4_h5_files=[]
for f in files_list:
  if f.find('ch4.h5') > 0:
    all_ch4_h5_files.append(f)
#This for loop confirms all the necessary files are in the
appropriate folder
for whichfile in range(12): #Using the13 files of  ch4.h5.
  h5_file =  target_dir+'//'+all_ch4_h5_files[whichfile]
  ABHD5_file = h5_file[:-7] +'_ch1.tif'
  DAPI_file = h5_file[:-7] +'_ch4.tif'
  #Making a test
  f1 = Path(DAPI_file)
  if not f1.is_file():
      print('DAPI file does not exist - ', DAPI_file)
  f1 = Path(ABHD5_file)
  if not f1.is_file():
      print('ABHD5 file does not exist - ', ABHD5_file)
  f1 = Path(h5_file)
  if not f1.is_file():
      print('H5 file does not exist - ', h5_file)
# To look if a point belongs inside a polygon of Voronoi
Diagram
def ray_tracing_method(x,y,poly):
    n = len(poly)
```

```
    inside = False
    p1x,p1y = poly[0]
    for i in range(n+1):
        p2x,p2y = poly[i % n]
        if y > min(p1y,p2y):
            if y <= max(p1y,p2y):
                if x <= max(p1x,p2x):
                    if p1y != p2y:
                        xints = (y-p1y)*(p2x-p1x)/(p2y-
p1y)+p1x
                    if p1x == p2x or x <= xints:
                        inside = not inside
        p1x,p1y = p2x,p2y
    return inside
# print(files_list)
for whichfile in range(13): # using the ch4.h5. Select any
intenger 0 through 12 for this variable.
  h5_file =  target_dir+'//'+all_ch4_h5_files[whichfile]
  ABHD5_file = h5_file[:-7] +'_ch1.tif'
  DAPI_file = h5_file[:-7] +'_ch4.tif'
#load tiff image
  d2 = tiff.imread(ABHD5_file)
  print("Working on file:", ABHD5_file)
  h5f = hh.File(h5_file,'r')
  LDtable = h5f['table']
  LDimages = h5f['images']
```

```
  numLD = LDtable.shape[0]##The shape attribute for numpy
arrays returns the dimensions of the array.
##If Y has n rows and m columns, then Y.shape is (n,m). So
Y.shape[0] is n.
#labling CM for the mask
  m=np.zeros(len(LDimages))
  b=np.zeros(len(LDimages))
  for i in range(len(LDimages)):
    m[i] = LDtable[i]['Bounding Box Minimum_0']#x
    b[i] = LDtable[i]['Bounding Box Minimum_1']#y
#labeling center of mass
  CM_all=[]
  for i in range(len(LDimages)):
    a=np.array(LDimages[str(i)]['labeling'])
    CM= ndimage.measurements.center_of_mass(a)
    CM_all.append(CM)
  CM_all= np.array(CM_all)
  ##Purple arrow (from (0,0) redbox to (0.0) black box )
  p_0=m
  p_1=b
  ##Purple arrow + red arrow ( from (0,0)redbox to CM )
  pO_0= CM_all[:,0]+p_0
  pO_1= CM_all[:,1]+p_1
#Plotting a Voronoi Diagram
  p=np.zeros((len(pO_0),2))
  p[:,0]=pO_0
```

```
  p[:,1]=pO_1
  vor = Voronoi(p)
#Make a mask for all polygons
  ##Make the masks from the DAPI data
  all_masks = [] ##All_masks is list of masks
  for region in vor.regions:
    mask = np.zeros((d2.shape))
    if not -1 in region:
      polygon = [vor.vertices[i] for i in region]
      if len(polygon) > 0:
        p = np.array(polygon)
        mini = np.max([int(np.floor(np.min(p[:,0])))-10,0])
        maxi =
np.min([int(np.ceil(np.max(p[:,0])))+10,mask.shape[0]])
        minj = np.max([int(np.floor(np.min(p[:,1])))-10,0])
        maxj =
np.min([int(np.ceil(np.max(p[:,1])))+10,mask.shape[1]])
        if maxj < analysis_range[0] or maxi <
analysis_range[0] or minj > analysis_range[1] or mini >
analysis_range[1]:
          continue
        for i in range(mini,maxi,1):#p[:,0]=X value
          for j in range(minj,maxj,1):#p[:,1]=y values
            if ray_tracing_method(i,j,polygon):
              mask[j,i] =1
```

```
all_masks.append(mask[analysis_range[0]:analysis_range[1],anal
ysis_range[0]:analysis_range[1]]) # each element of
"all_masks" is a list of mask
d3=d2[analysis_range[0]:analysis_range[1],analysis_range[0]:an
alysis_range[1]]
#Characterize "cells" of the tiff image of ABHD5
  means = np.zeros(len(all_masks))
  stds = np.zeros(len(all_masks))
  areas = np.zeros(len(all_masks))
  for masknow in range(len(all_masks)):
    if np.sum(all_masks[masknow])>0:
      dnow = d3*all_masks[masknow]
      stds[masknow] = np.std(dnow)
      areas[masknow] = np.sum(all_masks[masknow])
      sumnum= np.sum(dnow)
      means[masknow] = sumnum/areas[masknow]
  keep= means>0
  stds=stds[keep]
  areas=areas[keep]
  means= means[keep]
  np.savez(ABHD5_file[:-4],
                means = means,
                stds = stds,
                vor=vor,
                all_masks= all_masks,
                areas = areas)
```

```
"""# **Plotting**"""
#The tiff image of ABHD5
plt.imshow(d3)
plt.show()
#The center of mass
plt.plot(pO_0,pO_1, '.')
#The Voronoi Diagram
fig = plt.figure()
voronoi_plot_2d(vor)
#The range of axes
plt.xlim([analysis_range[0],analysis_range[1]])
plt.ylim([analysis_range[1],analysis_range[0]])
plt.show()
#Place a colored Voronoi diagram over the ABHD5 tiff image
plt.plot(vor.points[:,0],vor.points[:,1],'.w')
plt.xlim([analysis_range[0],analysis_range[1]])
plt.ylim([analysis_range[1],analysis_range[0]])
plt.imshow(d2)
#Colorize
for region in vor.regions:
   if not -1 in region:
        polygon = [vor.vertices[i] for i in region]
        plt.fill(*zip(*polygon), alpha=0.4)
#d3=d2[analysis_range[0]:analysis_range[1],analysis_range[0]:a
nalysis_range[1]]
```

```
        plt.xlim([analysis_range[1],analysis_range[0]])
        plt.ylim([analysis_range[1],analysis_range[0]])
plt.xlim([analysis_range[0],analysis_range[1]])
plt.ylim([analysis_range[1],analysis_range[0]])
plt.show()
#Histogram for Characterize " cells" of the ABHD5 tiff image
for loop
for whichfile in range(13): #Using the ch4.h5. Select any
intenger 0 through 12 for this variable.
  h5_file =  target_dir+'//'+all_ch4_h5_files[whichfile]
  ABHD5_file = h5_file[:-7] +'_ch1.tif'
  DAPI_file = h5_file[:-7] +'_ch4.tif'
  f=np.load(ABHD5_file[:-4]+'.npz')
  means=f['means']
  areas=f['areas']
  stds=f['stds']
  fig = plt.figure(figsize=(15,4))
  ax = fig.add_subplot(1,3,1)
  ax.hist(areas, 25)
  ax.plot([500,500],[0,29],':k')
  ax.plot([7000,7000],[0,29],':k')
  ax.set_xlabel('Cell Area (pixels)')
  ax.set_title(ABHD5_file[26:-4])
  ax.set_ylim([0,29])
  keep = np.all([areas>500,areas<7000],axis=0)
  ax = fig.add_subplot(1,3,2)
```

```
  ax.hist(means[keep],25)
  ax.set_xlabel('Mean brightness per cell')
  ax.set_title('Analysis of ABHD5 under DAPI-Voronoi masks')
  ax = fig.add_subplot(1,3,3)
  ax.hist(stds,25)
  ax.set_xlabel('Brightness standard deviation')
  plt.tight_layout()
  plt.savefig(ABHD5_file[:-4]+'_hists.jpg')
  plt.show()
masknow = int(len(all_masks)/16) # make choose any number less
than len(all_masks)
d= tiff.imread(DAPI_file)
fig = plt.figure(figsize=[15,8],dpi=150)
ax = fig.add_subplot(2,3,1)
ax.set_title('DAPI image')
plt.fill(*zip(*polygon), alpha=0.4)
d0=d[analysis_range[0]:analysis_range[1],analysis_range[0]:ana
lysis_range[1]]
##ax.imshow(d0) for see image with caler
ax.imshow(d0,cmap='gray')
#This is the mask=0 and the code finds the pixel inside
#If a point is inside the polygon،the code will change the
value in pixels to one (e.x mask[900:1000,:]=1 I choose which
values equal 1 .)
ax = fig.add_subplot(2,3,2)
ax.imshow(all_masks[masknow],cmap='gray')
```

```
#Demonstrate multiplying a mask by the TIF image
ax = fig.add_subplot(2,3,3)
d0=d[analysis_range[0]:analysis_range[1],analysis_range[0]:ana
lysis_range[1]]
m = d0 * all_masks[masknow] #The TIF image is now Zero exept
under the mask(so you can see what under onle in the part not
0)
ax.imshow(m)
ax = fig.add_subplot(2,3,4)
ax.imshow(d3)
ax.set_title('ABHD5 image')
ax = fig.add_subplot(2,3,5)
ax.imshow(d3*all_masks[masknow],cmap='gray')
plt.tight_layout()
plt.savefig(ABHD5_file[:-4]+'_'+str(masknow)+'.jpg')
plt.show()

"""# **Show similarity and differences between repeats the
files with the same name but different numbers**"""
#4 different conditions:
#1-'fsk 3hr_13342','fsk 3hr_13340','fsk 3hr_13339','fsk
3hr_13341'
#2-' dmso_13336',' dmso_13338',' dmso_13337'
#3-'liso 3hr_13344','liso 3hr_13343','liso 3hr_13345'
#4-'162 3hr_13347' ,'162 3hr_13348','162 3hr_13346'
whichfiles_list = [0,2,5,8] # for condition 1(fsk)
```

```
#whichfiles_list = [3,4,10]  # for condition 2(dmso)
#whichfiles_list = [1,6,11]  # for condition 3(lso)
#whichfiles_list = [7,9,12] # for condition 4(162)
#The all files in all condeations
lab =['fsk 3hr_13342','liso 3hr_13344',
      'fsk 3hr_13340','dmso_13336','dmso_13338',
      'fsk 3hr_13339','liso 3hr_13343','162 3hr_13347',
      'fsk 3hr_13341','162 3hr_13348','dmso_13337','Iso
3hr_13345','162 3hr_13346']
col = 'rgbm'
fig = plt.figure(dpi=120)
bins = np.linspace(0,2500,50) ##Return evenly spaced numbers
over a specified interval(start , stop , Number of samples to
generate)
##Defining a list in the range(0,2500) with 50 sample values.
binx = (bins[1:]+bins[:-1])/2 #bins[1:] includes all the
values in the list except 1st index item.
# bins[:-1] includes all the values in the list bins except
the last index /2 to see the center
for i,whichfile in enumerate(whichfiles_list): #Enumerates
runs a loop with a counter alongside
  h5_file =  target_dir+'//'+all_ch4_h5_files[whichfile]
#Basic string concatenation and storing ultimate string into
h_file
  ABHD5_file = h5_file[:-7] +'_ch1.tif' #Includes all the
values in the list h_file till 7th index from the last and
```

```
then string concatenation
  print(lab[whichfile],' corresponds to ', ABHD5_file)
#Choosing whichfile index from the lab list and print three
values in a space separated manner
#loads pickled objects from .npz files, filepath defined by
ABHD5_file[:-4]+'.npz'
#which includes all the values till 4th index from the last
and concatenated '.npz' extension.
  f = np.load(ABHD5_file[:-4]+'.npz')
  means = f['means']
  f.close()
#Create a histogram, bins add the int or sequence of scalars,
which is optional
  H = np.histogram(means,bins=bins)
  print(len(means))
  plt.plot(binx,H[0],
           '-'+col[i],
           label=lab[whichfile], linewidth=2) #Plot the graph
with the points defined in the binx list
plt.legend()
plt.ylabel('Number of cells')
plt.xlabel('Brightness per cell')
#plt.show()
"""# **Show how the different conditions result in different
data**"""
import matplotlib.pyplot as plt
```

```
import numpy as np
whichfiles_list2 = ([3,4,10],  # for condition 1(dmso)
                    [0,2,5,8], # for condition 2(fsk)
                    [1,6,11],  # for condition 3(lso)
                    [7,9,12]) # for condition 4(162)
means4 = [] #list of conditions
[0,1,2,3]which[dmso,fsk,iso,162]
areas4=[]
stds4=[]
for i in range(len(whichfiles_list2)):
  whichfiles_list = whichfiles_list2[i] #The list of every
condations
  means_temp = []
  areas_temp = []
  stds_temp = []
  for whichfile in whichfiles_list:
    h5_file =  target_dir+'//'+all_ch4_h5_files[whichfile]
#Basic string concatenation and storing ultimate string into
h5_file
    ABHD5_file = h5_file[:-7] +'_ch1.tif' #Includes all the
values in the list h5_file till 7th index from the last and
then string concatenation
    f = np.load(ABHD5_file[:-4]+'.npz')
    means = f['means'] #Test the shape of (f =3 items)
    areas=f['areas']
    stds=f['stds']
```

```
    keep=np.all([areas>1000,areas<5000],axis=0)
    means=means[keep]
    areas=areas[keep]
    stds=stds[keep]
    areas=areas[means >0]*0.36
    stds=stds[means >0]**2
    means= means[means >0]
    f.close()
    means_temp.append(means)  #This should be a list of 3 or 4
"means", one from each image/file; test the shape of this
    areas_temp.append(areas)
    stds_temp.append(stds)
  means4.append(np.concatenate(means_temp)) #This will
eventually have 4 entries, one from each condition; test the
shape of this
  #Temp is a list of every image by combining together these
images for each condition
  #means4 is all data from each files
  areas4.append(np.concatenate(areas_temp))
  stds4.append(np.concatenate(stds_temp))
# Plotting swarm plot inside the boxplot
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
data =[means4,areas4,stds4]
title= ['brightness','areas','brightness variance']
```

```
ylab= ['Intensity (a.u)' , 'Area (µm^2)','Brightness Variance
(a.u)']
for i in range(3):
  fig = plt.figure(figsize = (5,4),dpi=150)
  fs = 12 # fontsize for axis labels
  fs_tick = fs
  lw_axis = 1.5
  matplotlib.rc('xtick', labelsize=fs_tick)
  matplotlib.rc('xtick.major',width=lw_axis,size=10)
  matplotlib.rc('ytick', labelsize=fs_tick)
  matplotlib.rc('ytick.major',width=lw_axis,size=10)
  matplotlib.rc('axes', linewidth=lw_axis)
  #Swarm plot
  ax =
sns.swarmplot(data=data[i],palette=sns.color_palette("pastel")
, zorder=0 , size=1.5)
#Boxplot
  sns.boxplot( data=data[i], notch= True,
                  showcaps=True,boxprops={'facecolor':'None'},

showfliers=False,whiskerprops={'linewidth':1}, ax=ax)
  plt.xticks([0,1,2,3],['DMSO' , 'FSK' , 'ISO' , '162'])
  ax.set_ylabel(' Single-Cell '+ ylab[i], fontsize=fs)
  matplotlib.pyplot.title('Single-cell ABHD5 '+title[i]+' upon
simulation' ,fontsize=fs)
  plt.show()
```

```
"""# P-value
I have three different sets of comparisons to make
"""
from scipy import stats
means_dmso=means4[0]
means_fsk=means4[1]
means_iso=means4[2]
means_162=means4[3]
print("Comparison1:(fsk - iso)")
print(stats.ttest_ind(means_fsk,means_iso,equal_var = False))
print("Comparison2(fsk - 162)")
print(stats.ttest_ind(means_fsk,means_162,equal_var = False))
print("Comparison3(iso - 162)")
print(stats.ttest_ind(means_iso,means_162,equal_var = False))
print("Comparison4(dmso - 162)")
print(stats.ttest_ind(means_dmso,means_162, equal_var =
False))
print("Comparison5(dmso - fsk)")
print(stats.ttest_ind(means_dmso,means_fsk, equal_var =
False))
print("Comparison6(dmso - iso)")
print(stats.ttest_ind(means_dmso,means_iso, equal_var =
False))
```

BIBLIOGRAPHY

[1] S. Venkatakrishnan and C. Kalyani, “Basics of image processing technologies,” *Int. J. World. Res*., vol. 1, no. 34, pp. 55–58, 2016.

[2] C. Solomon and T. Breckon, *Fundamentals of digital image processing: A practical approach with examples in MatLab*. John Wiley & Sons, p. 355, 2011.

[3] S. Berg, D. Kutra, T. Kroeger, C. N. Straehle, B. X. Kausler, C. Haubold, M. Schiegg, J. Ales, T. Beier, M. Rudy, K. Eren, J. I. Cervantes, B. Xu, F. Beuttenmueller, A. Wolny, C. Zhang, U. Koethe, F. A. Hamprecht, and A. Kreshuk, “Ilastik: Interactive machine learning for (bio)image analysis,” *Nat. Meth*., vol. 15, no. 12, pp. 1226–1232, 2019. https://doi.org/10.1038/s41592-019-0582-9.

[4] C. Sommer, C. Straehle, U. Köthe, and F. Hamprecht, “Ilastik: Interactive learning and segmentationtoolkit,” vol. 2011, pp. 230–233, 2011. https://doi.org/10.1109/ISBI.2011.5872394.

[5] Ilastik - Overview https://www.ilastik.org/documentation/index.html (accessed 2022 -03 -04).

[6] M. A. Sanders, F. Madoux, L. Mladenovic, H. Zhang, X. Ye, M. Angrish, E. P. Mottillo, J. A. Caruso, G. Halvorsen, W. R. Roush, P. Chase, P. Hodder, and J. G. Granneman, “Endogenous and synthetic ABHD5 ligands regulate ABHD5-perilipin interactions and lipolysis in fat and muscle,” *Cell. Metab.*, vol. 22, no. 5, pp. 851–860, 2015. https://doi.org/10.1016/j.cmet.2015.08.023.

[7] R. Pal, M. Mamidi, A. Das, and R. Bhonde, “diverse effects of dimethyl sulfoxide (DMSO) on the differentiation potential of human embryonic stem cells,” *Arch. Toxicol*., vol. 86, pp. 651–661, 2011. https://doi.org/10.1007/s00204-011-0782-2.

[8] S. Tunçer, R. Gurbanov, I. Sheraj, E. Solel, O. Esenturk, and S. Banerjee, “low dose dimethyl sulfoxide driven gross molecular changes have the potential to interfere with various cellular processes,” *Sci. Rep.*, vol. 8, no. 1, p. 14828, 2018.. https://doi.org/10.1038/s41598-018-33234-z.

[9] J. G. Granneman, H. P. H. Moore, R. Krishnamoorthy, and M. Rathod, “Perilipin controls lipolysis by regulating the interactions of ab-hydrolase containing 5 (abhd5) and adipose triglyceride lipase (atgl) *,” *J. Biol. Chem*., vol. 284, no. 50 pp. 34538–34544, 2009. https://doi.org/10.1074/jbc.M109.068478.

ABSTRACT

COMPUTATIONAL SINGLE-CELL ANALYSIS OF CONFOCAL FLUORESCENCE IMAGES WITH DAPI-GENERATED MASKS

by

MUNIRAH AL-DUHAILAN

May 2022

Advisor: Dr. Christopher V Kelly

Major: Physics

Degree: Master of Science

Lipolysis is a metabolic pathway in which free fatty acids are mobilized from stored triglycerides. The rate-limiting enzyme in this process is adipose triglyceride lipase, which is regulated by α/β-hydrolase domain-containing protein 5 (ABHD5) via both natural and synthetic pathways. With advanced artificial neural networks, image processing methods can extract quantitative results from fluorescence images. The segmentation of complex biological images, in which regions of the image are labeled as distinct masks, is the first step in image analysis. Ilastik, a machine-learning software, performs image segmentation with a user-trained neural network and custom key feature labels. The software's results are evaluated using a custom Python script, resulting in a new workflow that incorporates Ilastik for the construction of single-cell data from confocal fluorescence images. We analyzed multi-color fluorescence images of tissues to determine the growth and metabolism of lipid droplets. Moreover, the use of neural-network-based fluorescence image analysis to measure single-cell triglyceride storage and ABHD5 expression upon stimulation with isoproterenol, SR3420, dimethyl sulfoxide, or forskolin is reported. We demonstrate enhanced quantitative information for hypothesis testing in the assessment of single-cell behaviors and metabolic pathways.

AUTOBIOGRAPHICAL STATEMENT

Since childhood, I have always had a great passion in knowing the physicists and their achievements until I realized that physics is the basis of the world's progress on the civil and cultural levels. This passion accompanied me until I graduated from high school and enrolled as a physics student in King Faisal University, Kingdom of Saudi Arabia. For me as a student, it was to acquire skills and to know more about physics branches which will ultimately make me a good candidate for any academic institute. I did it relentlessly and I enjoyed every challenge!

Taught by excellent professors who encouraged students to do things passionately and differently, I learned to measure, to experiment, to test, to interpret and to write reports in an accurate and unique way. One of my researches was chosen by my professor to be a reference for most of the students. My professors were impressed by my graduation research. The university-based on their recommendation- invited me to present it in the Research Day held for faculty professors and masters and PH candidates. It was about the positive effects of radioactive elements on plants and it supervised by Dr. Amal Hamdan.

After graduating from college, I was appointed as a teaching assistant of physics in King Saud bin Abdulaziz University for Health Sciences. The university offered me a scholarship to complete a master's and doctorate degrees. A master's program in General physics or Biophysics.

During my master's at Wayne State University, I had opportunities to acquire the computing technique in grid computing systems as well as software skills with a good understanding of the general Physics courses and Biophysics. I hope to have opportunities to stude Phd at Wayne State University and continue work with my Dr Christopher V Kelly in Lipid Droplet Interest Group.

Upon having the master's and doctorate degrees, I will employ the skills and information I obtain to develop a smart generation of physicists who look to produce science. Female students will have a considerable amount of my attention and care. I will encourage them to specialize in physics in general and in Biophysics in particular to pave the way for equal opportunities with male students. I will, also, work with other members of faculty to establish the department of Biophysics.