University of Arizona

College of Engineering

ENGR 498B: Cross-Disciplinary Design

Section 2

Final Report

Date: May 1, 2019

Team 18075

Diego Alcantara

Pedro Alcaraz

Andrew Burger

Xinyu “Emma” Li

Adriana Stohn

Table of Contents

Scope and Introduction	1
System Block Diagram	2
Technical Data Package	3-22
Acceptance Test Procedure	23-24
Models and Analyses	25-27
Acceptance Test Results	28-31
Final Budget	32
Lessons Learned	32-33
Appendix	34-36

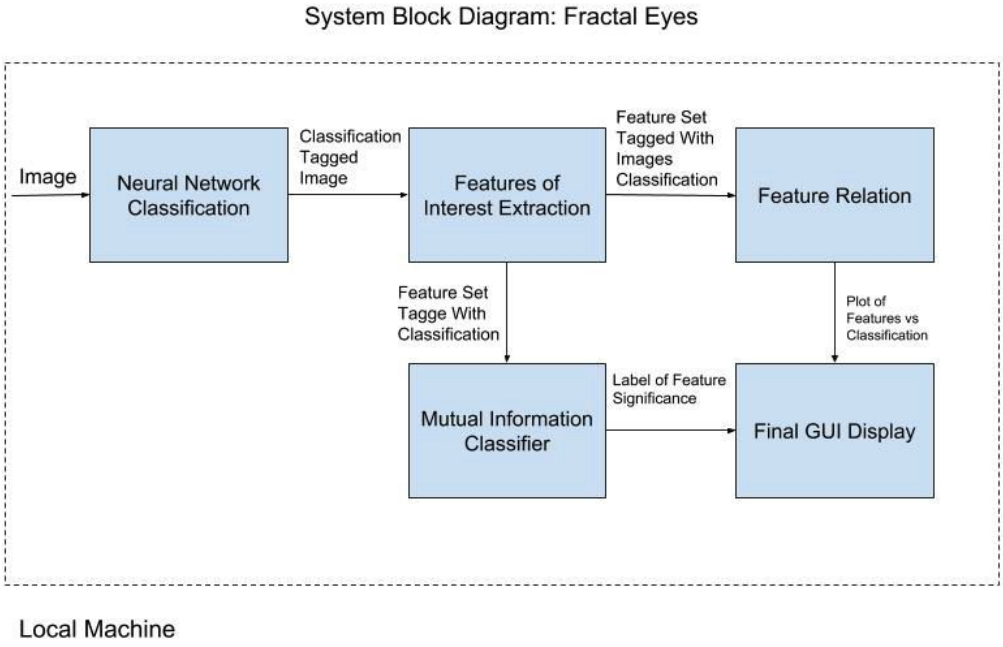
Scope and Introduction

This document consists of all of the planning and technical development done by Team 18075 towards the fruition of the Fractal Eyes medical image processing tool over the past school year culminating in completion and presentation on April 29th, 2019. The Fractal Eyes tool was designed to combat the propensity for error in medical image analysis resulting from the qualitative image assessment by the human eye. The tool exploits the robust quantitative analysis capabilities of the common desktop computer to perform basic image analytics, such as feature extraction and mutual information, and to levy advanced classification algorithms, such as neural network frameworks.

The Fractal Eyes tool offers what no other known medical image processing tool can because it offers metrics about to what degree a certain image feature correlates to another within a single dataset. The development of this functionality could pave the way for innovative mutual information based image processing in the medical community, a paradigm shift that has yet to happen.

The Technical Data Package portion of this document details the system architecture as well as the primary code functions that makes up the analysis of the Fractal Eyes tool.

System Block Diagram



University of Arizona

College of Engineering

ENGR 498B: Cross-Disciplinary Design

Section 2

TDP

Date: January 15, 2019

Team 18075

Diego Alcantara
Pedro Alcaraz
Andrew Burger
Xinyu “Emma” Li
Adriana Stohn

Date	Description	By
1/15/19	Initial Release	DA
2/14/19	Removed Irrelevant Features	AS/DA
2/19/19	Filled in Table S1 Values	DA
4/16/19	Revised sections to reflect final product	DA

Table of Contents

Introduction	4
System Description	5-6
System Architecture and Requirements Traceability	7-8
Prediction of Performance Margins and Accuracy	9
Software Design Document	10-22
Acceptance Test Procedures	23-24

Introduction

Medical imaging is a critical component of disease diagnosis in modern medical practice. Although advances in medical imaging technologies have enabled healthcare providers to implement more effective point of care strategies, the analysis of medical images remains inefficient and highly subjective.

For example, radiologists rely almost exclusively on the visual inspection of clinically acquired images to diagnose abnormal phenomena in patient tissues. As a result, any radiologist's analysis of a medical image is liable to biased influences from the radiologist's own convictions or professional background, leading to inconsistencies in medical diagnoses. In computer science, the field of image processing utilizes a wealth of mathematical algorithms and processing routines to perform quantitative analyses on visual data in the form of images.

Incorporating image processing in medical imaging can allow for an quantitative analysis of medical image data, begetting more reliable and consistent diagnoses.

Here, a medical image detail extraction and analysis tool, Fractal Eyes, is described. The Fractal Eyes tool is proof of concept model to mediate the development of more robust medical image analytics, such as natural language processing of quantitative feature analysis.

System Description

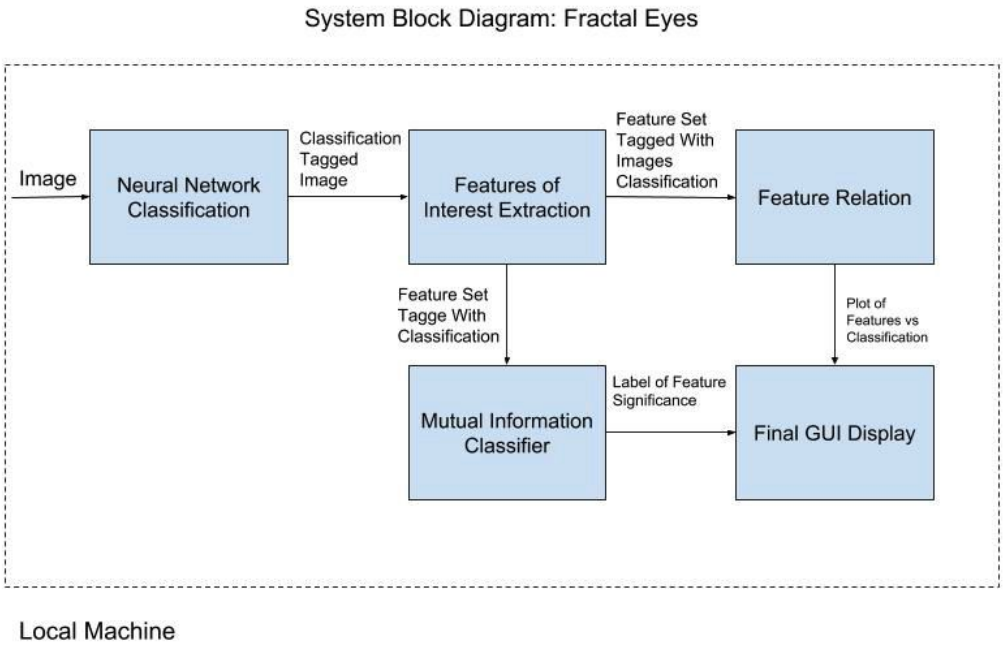


Figure 1: Fractal Eyes System Block Diagram

The Fractal Eyes system is designed to find correlative relationships between features within an image and their corresponding classification label.

The initial subsystem will classify an input image with class label 1 or class label 2 using a convolutional neural network. This classification will be tied to all features extracted from the image further in the pipeline.

The second subsystem will extract features of interest from the input image. The features of interest in this application of the technology will be: average color of the image from RGB channels, number of features extracted through the automatic feature extraction process (this is a sum of blob/circular-like features extracted through a Difference of Gaussian Algorithm, Laplacian of Gaussian Algorithm, and Determinant of Hessian Algorithm), the average size of the features extracted through the automatic feature extraction process, and the overall luminosity value of the image.

The Feature Relation subsystem will generate pair plots comparing each feature to one another. The plots will be used as a visual justification for the classification applied to the image.

The Mutual Information Classifier subsystem will compute the mutual information scores between the features to quantitatively determine which features are significant in the attachment of the applied classification label.

The Final GUI Display subsystem will communicate all information gathered during the analysis process to the user (i.e. pair plots, classification label, mutual information scores, etc.)

System Architecture and Requirements Traceability

System Architecture:

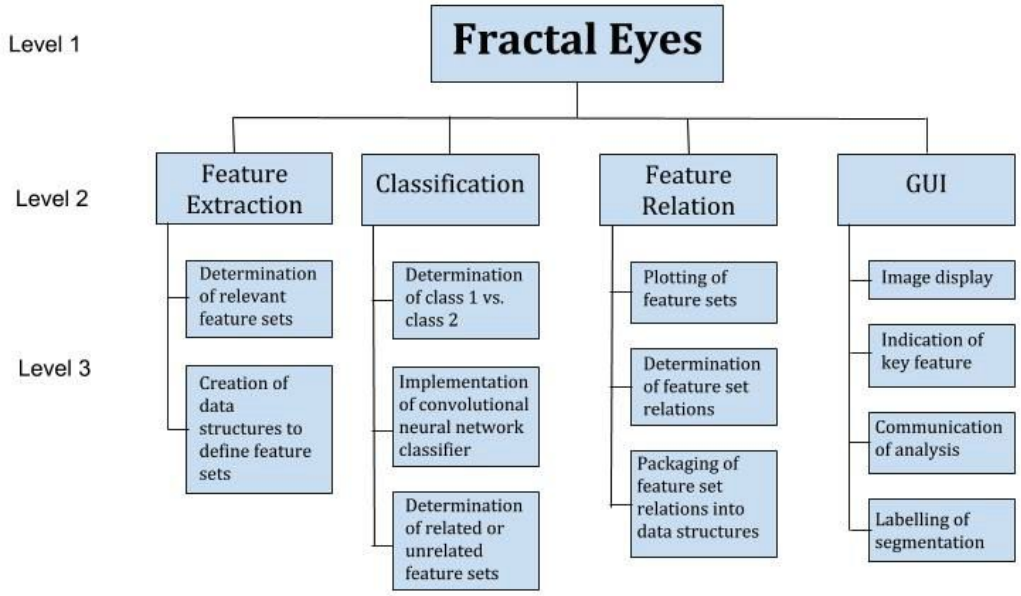


Figure 2: Fractal Eyes System Architecture

Requirements Traceability to Sub-Systems:

Table 1: Fractal Eyes Traceability Matrix											
System Requirement	VM	Sub-Assembly									
		Neural Network Classification	VM	Features of Interest Extraction	VM	Feature Relation	VM	Mutual Information Classifier	VM	Final GUI Display	VM
4.1.1 MRI or Histological Images	D	1.0 direct flow	D								
4.1.2 Image dimension invariant	T										
4.2.1 Automate feature extraction	D			1.0 direct flow	TD						
4.2.2 Features of interest extraction	TD			2.0 direct flow	TD						
4.3.1 Image visual display	D									1.0 direct flow	D
4.3.2 Features of interest visual display	D									2.0 direct flow	D
4.3.3 Run on desktop environment	D									3.0 direct flow	D
4.4.1 Find relationships between features	D					1.0 direct flow	D	1.0 direct flow	D		
4.5.1 Plot relationships between features	D					2.0 direct flow	D	2.0 direct flow	D		

Prediction of Performance Margins/Accuracy

The only sub-system of our project that requires a performance metric is the initial Neural Network Classifier. Our estimated performance for our neural network is 80%.

Neural networks tend to do much better with large amounts of data (many samples). However, our dataset is confined to 900 samples of each class. In most cases this would not be enough samples to attain good performance. However, the two classes are visually very different from one another which will increase performance despite the low sample size.

Because these two classes are so different and because we are using a binary classifier, the worst performance we could obtain is 50% (which would be considered random guessing). Despite the low sample size, 80% is a realistic assumption. In addition, for this application of the technology, accuracy is low risk and thus we would not need to set it particularly high.

University of Arizona

College of Engineering

ENGR 498B: Cross-Disciplinary Design

Section 2

SDD

Date: January 15, 2019

Team 18075

Diego Alcantara
Pedro Alcaraz
Andrew Burger
Xinyu “Emma” Li
Adriana Stohn

Table of Contents

1.0 Scope **12**
1.1 Identification 12
1.2 System Overview 13
1.3 Doc Overview 13

2.0 Referenced documents **13**

3.0 CSCI-wide design decisions **13-14**
3.1 Class1/Class2 Classification 13
3.2 Features of Interest Extraction 13
3.5 Feature Relation 13
3.6 Mutual Information Classifier 13-14

4.0 CSCI Architectural Design **14**

5.0 CSCI Detailed Design **14-22**
5.1 Graphical User Interface 14-17
Fig. 1: GUI display demonstrating image upload and analysis features via user input command. 15
Fig. 2: GUI displays feature vs feature plots. 15
Fig. 3: GUI displays the selected feature vs. feature plot by user command. 16

5.2 Pseudo Code for Subsystems 17-22
Table S1. Features of Interest Extraction Verification 19

6.0 Notes **22**

1.0 Scope

1.1 Identification

This system SDD gives the design of the software CSCI for the Classification, Feature Relation, Mutual Information, and Graphic User Interface Subsystems. It defines the software modules required to accurately classify image datasets and generate feature to feature and feature to class relationships.

1.2 System Overview

The purpose of this system is to demonstrate the potential impact of automated image processing and its possible use in medical industry. The FractalEyes™ tool will allow for a quantitative analysis of image data sets; where, machine learning algorithms will enable automation of complex image processing routines. A novel characteristic of the FractalEyes™ Image processing suite is its ability to generate relevant feature to feature relations; where, the tool will not only classify image sets and extract their relevant features but also determine which features were most relevant to the classification of an image as well as generate relations between the extracted features sets.

This system will solely be developed utilizing software. The system will be composed of five (5) major subsystems each with unique functionality to allow for the aforementioned image processing capabilities; these include an a (1) Classification Subsystem (C), a (2) Feature of Interest Extraction Subsystem (FIE), a (3) Feature Relation Subsystem (FR), a (4) Mutual Information Subsystem (MI), and a (5) Graphic User Interface Subsystem (GUI).

The FEI subsystem will serve as the modality for feature extraction from image sets. These subsystems will feature automated feature extraction algorithms to allow for analysis of an assortment of image sets with varying elements of interest. The C subsystem will be developed utilizing a neural network. This will allow for efficient and effective classification of images within image sets. The FR and MI subsystems will primarily be used to develop inter-feature relations as they pertain to the classification of the images analyzed and subsequently generate a general hierarchy of features based on the the overall importance of individual features to the assigned image classification.

Dr. Marvin Slepian is the project sponsor for the development of the FractalEyes™Tool. The developers of the FractalEyes tool include Diego Alcantara, Pedro Alcaraz, Andrew Burger, Xinyu “Emma” Li, and Adriana Stohn. Paperwork concerning the tool as well as the source code will be provided to the project sponsor in the final project package. The operating site for this project will be a desktop environment capable of running the delivered program.

1.3 Doc Overview

This SDD gives the design of software CSCI for the Fractal Eyes system. It indicates the software modules required to segment the image and extract the feature, and firstly classify the image as class 1 or class 2, then calculate feature relatedness. Finally, it is required to display the plot of relatable feature sets on the screen. The major components are Neural Network Classification, Feature of Interest Extraction, Feature Relation and Mutual Information Classification, and Graphical User Interface (GUI).

2.0 Referenced documents

Open Source Computer Vision Library (OpenCV) - https://docs.opencv.org/
The Python Deep Learning Library (Keras) - https://keras.io/
Statistical Data Visualization (Seaborn) - https://seaborn.pydata.org/
Python Data Analysis Library (Pandas) - https://pandas.pydata.org/
Parallelized Mutual Information based Feature Selection Module (MIFS) - https://github.com/danielhomola/mifs
Numerical Data Analysis (NumPy) - http://www.numpy.org/

3.0 CSCI-wide design decisions

3.1 Class1/Class2 Classification

This CSCI shall be able to classify images as Class1 or Class2 using a neural network framework.. The CSCI will store the resulting internal weights, and when the user passes in an unlabelled image, the CSCI will label that image as either Class1 or Class2.

3.2 Features of Interest Extraction

This CSCI shall be able to automatically extract features of interest and store the information for analysis. This CSCI shall store coordinates of features and store the coordinates in a data structure (vector, dictionary, array, etc.).

3.5 Feature Relation

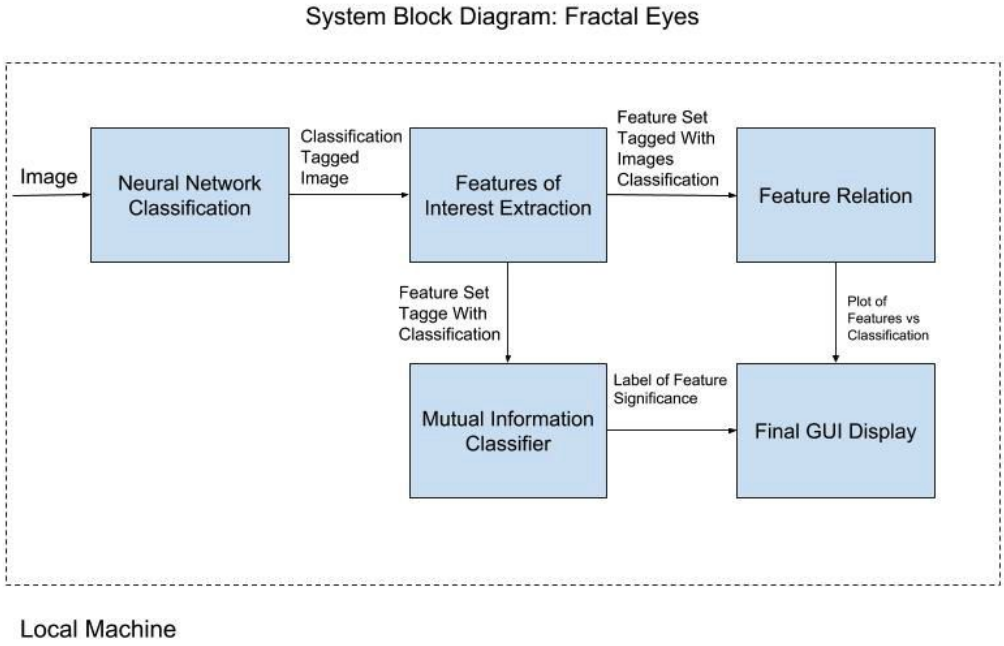
The CSCI shall be able to display feature by feature plots to show the user relationships between features. Also the CSCI shall give users a visual display of which features are significant via histograms.

3.6 Mutual Information Classifier

The CSCI shall be able to measure the joint mutual information between features in the image set and the corresponding labels attached to those features. The joint mutual information function will

then be required to filter out the lowest joint mutual information scores via a user defined threshold to create a final list of significant features.

4.0 CSCI Architectural Design



5.0 CSCI Detailed Design

5.1 Graphical User Interface

The *GUI CSC* serves as Graphical User Interface for users and display the data/feature information within images. It displays the feature by feature plots, Class1/Class2 classification label, and correlative data and explanation.

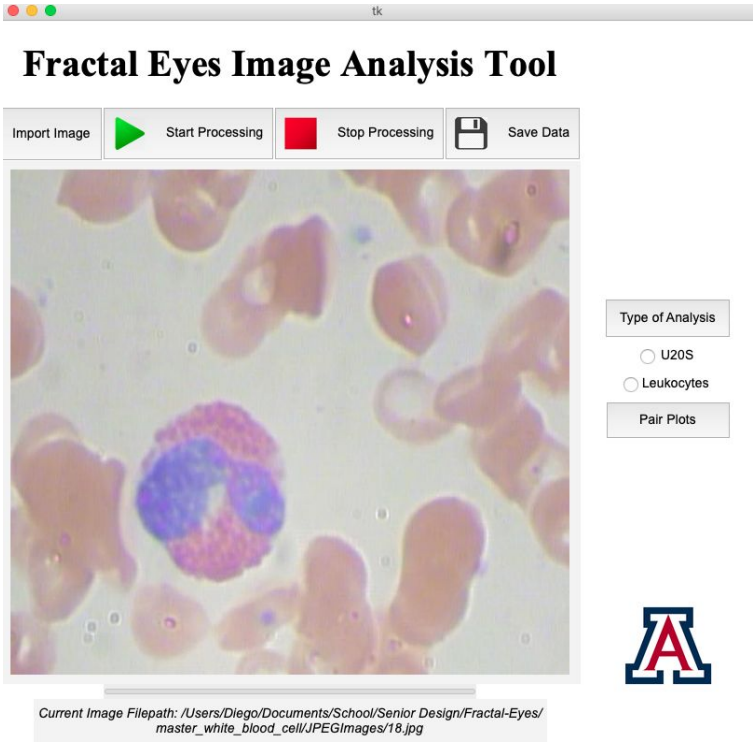


Fig. 1: GUI display demonstrating image upload and analysis features via user input command (Green triangle: start, Red square: stop, Black Floppy Disk: save).

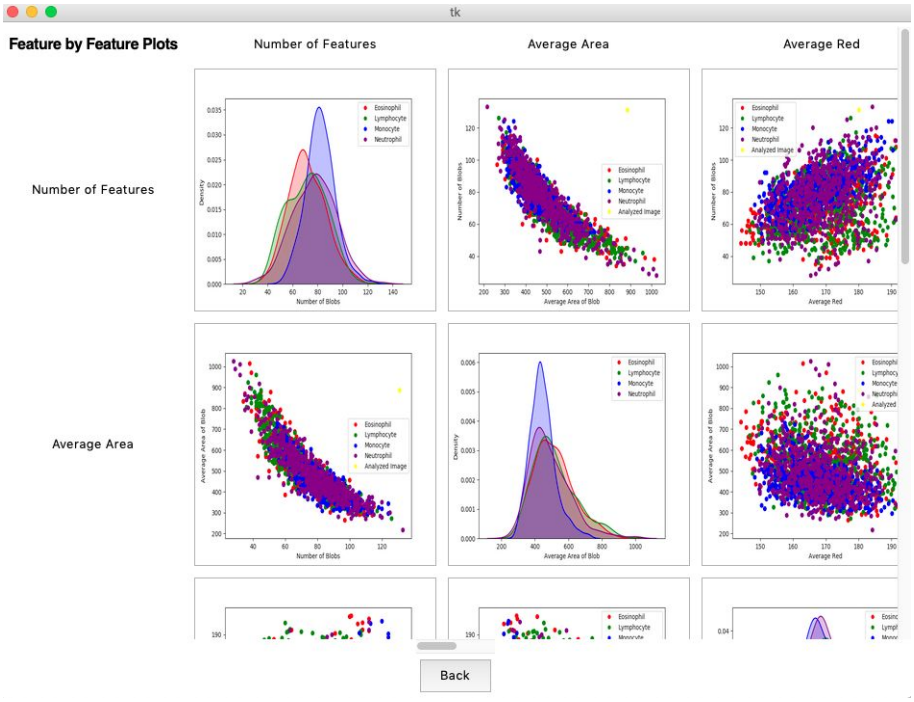


Fig. 2: GUI displays feature vs feature plots.

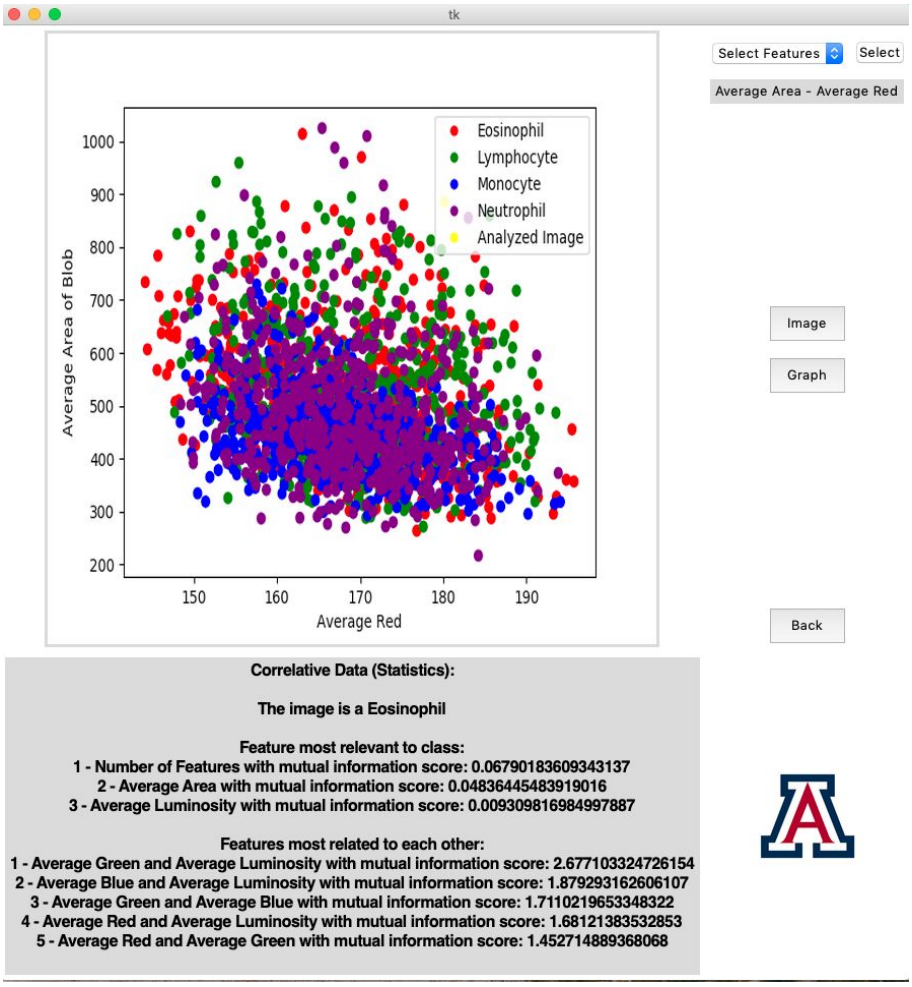


Fig. 3: GUI displays the selected feature vs. feature plot and correlative data by user command

5.2 Pseudo Code for Subsystems

5.2.1 Neural Network Classification

```
from keras.layers import Dense, Dropout, Flatten, Activation
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.layers import Lambda
from keras.models import Sequential
import pandas as pd
from keras.utils import np_utils
import numpy as np
from keras.preprocessing.image import ImageDataGenerator, array_to_img,
img_to_array, load_img
from PIL import Image
from keras import backend as K
from keras import optimizers

model = Sequential()
Imagedimensions = x ,y

# establish variables for training network

Epochs, batch_size, num_trainsamples, num_validation samples

#check image channel orientation (LWH, HWL) to setup input shape

if(format == LengthWIdthHeight)
        Adjust input_shape accordingly
Else
        Adjust input_shape accordingly

#Setup convolution network architecture
model.add(Conv2D(32,activation="relu"))
model.add(Conv2D(64,activation="relu"))
model.add(Dense(64))
model.add(Dense(NUMBER_CLASSES,activation="softmax"))

#Compile model
model.compile(loss='binary_crossentropy/categorical_crossentropy',
optimizer='rmsprop', metrics=['accuracy'])
   #adam is efficient gradient descent algorithm
```

```
#Train model

#generate training data
Train_datagen = ImageDataGenerator

#generate test data
Test_datagen = ImageDataGenerator

#Train using flow from directory
Train_generator = train_datagen.flow_from_directory(args)
#test using  flow from directory
Validation_generator = test_datagen.flow_from_directory(args)

#Now fit the model
model.fit_generator(args)
#save model
model.save("model_name")

#Evaluate model
   #Statistical analysis to assess accuracy of model

#Predict labelling of unlabeled images
unknownImage = np.ones((300,300))
predictedLabel = model.predict(unknownImage)
   #predictedLabel will indicate whether the image is class1 or class2
```

5.2.2 Features of Interest Extraction

Table S1. Features of Interest Extraction Verification

A-priori Testable Features	Expected Values		Experimental Values
	Obtained by	Values	
Average Color	Third-party image processing software, ImageJ	U2OS [R,G,B] = [12, 12, 12] White Blood Cell[R,G,B] = [190, 170, 180]	U2OS [R,G,B] = [15, 15, 15] White Blood Cell [R,G,B] = [185, 130, 180]
Number of Extracted Features	Visual inspection	[U2OS, White Blood] = [300, 200]	[U2OS, White Blood] = [300, 125]
Average Area of Feature Extracted (# of Pixels)	Visual inspection	[U2OS, White Blood] = [150, 400]	[U2OS, White Blood] = [200, 750]
Intensity	Third-party image processing software, ImageJ	[U2OS, White Blood] = [10, 320]	[U2OS, White Blood] = [25, 180]

5.2.2.1 Average color

```
#import io library
From skimage import io

# load in the image
img = io.imread('filepath')

# call mean function

average = img.mean(axis=0).mean(axis=0)
```

5.2.2.2 Finding average exposure of image

```
#import scikit image library exposure
From skimage import exposure

#get histogram data of image which is an Array of data
hist= exposure.histogram(image,nbins)
```

```
#Average histogram data to find average pixel intensity

For i in range(hist data)
        sum = sum + hist data(i)

Mean = sum/len(hist data)
```

5.2.2.3 Automated Feature Extraction

```
#define parameters for Difference of Gaussian functions for each cell class (see code
                                                                    for example)
Image = rb2gray(image) #convert to grayscale
log_feats = blob_log(image, parameters…) #computes Laplacian of Gaussian features
dog_feats = blob_dog(image, parameters…) #computes Difference of Gaussian
                                                                    features
doh_feats = blob_doh(image, parameters..) #computes Determinant of Hessian
                                                                    features
For each of the above features
        blobs[:,2] = blobs[:,2] * sqrt(2) #computes radius and puts radii in third
                                                            column of matrix
total_blobs = len(log_feats) + len(dog_feats) + len(doh_feats)

For each of the above features
        compute area
avg_area = sum(areas) / len(blobs)
```

5.2.5 Feature Relation

Feature by feature plot

```
import pandas as pd
import seaborn as sns

X = arrayofFeatures + labels
# this must be a numpy array structure of array should be a 2D array with image
        indices as the rows, feature values along each column and the class label as
                                                                    the last column

# create a DataFrame to input to pairplot function
DataFrame = pd.DataFrame(X)

DataFrame.columns = [‘Feature1’, ‘Feature2’, ‘Class1/Class2’]
```

```
# these are the titles of each column

Plot =  seaborn.pairplot(DataFrame)
```

5.2.6 Mutual Information Classifier

Joint Mutual Information classifier

```
# First will need to compute the mutual information between a continuous feature
# vector x and discrete random variable y (class label)

x = vector of feature values for a single feature
y = discrete label vector # i.e. ([0,1,0,1])
k = number of neighbors in the nearest neighbor classifier
# part of computing mutual information requires using KNN classifier

MI = JMI_compute(x,y,k)
features = list of all features

Mutual_information_scores = []
for i in range(len(features)):
        MI = JMI_compute(feature[i],y,k)
        Mutual_information_scores.append(MI)
# This creates  a list of mutual information scores of each feature

#Now select based on a user designed threshold what features are relevant
threshold = userdefined

relevant_features = []
for i in range(len(Mutual_information_scores)):
        if Mutual_information_scores[i] > threshold:
                relevant_features.append(Mutual_information_scores[i])

# at this point relevant_features is a final list of features that matter
```

5.2.8 Full Image Set Analysis

```
Declare number of samples

for i in range(0, num_samples):
        extract features for each image in test database
        append to list of each individual features

create vector for class labels
```

apply appropriate class label (i.e. 0,1,2,3...n) to each index corresponding to the image examined at given index
array_of_feats = numpy.column_stack((feature_1, feature_2,... feature_n, labels))
numpy.savetxt("filepath.csv", array_of_feats, delimiter = ',')
#this creates a csv file with feature values along the columns and the class label in the last column

5.2.8 Image Pipeline

class_label = make call to neural network predictor
num_blobs, avg_area_of_blobs = make call to full blob analysis function
avg_color = make call to average color function
lum_avg = make call to average luminosity function

create array for data in the form
[num_blobs, avg_area, avg_color_red, avg_color_blue, avg_color_green, lum_avg, class_label]
make call to pair plotting()
class_labels = ["class_one", "class_two"]
feat_names = ["Number of Features", "Average Area", "Average Red", "Average Green", "Average Blue", "Average Luminance", "Class"]
data = numpy.loadtxt(full_image_set_analysis)
dataframe = pandas.DataFrame(data, columns=feat_names)
mutual_info = make call to mutual information function
top_n_features = sort features by mutual information score and return top n

make call to natural language explanation

6.0 Notes

This section left intentionally blank

University of Arizona

College of Engineering

ENGR 498B: Cross-Disciplinary Design

Section 2

ATP

Date: January 15, 2019

Team 18075

Diego Alcantara

Pedro Alcaraz

Andrew Burger

Xinyu “Emma” Li

Adriana Stohn

5.1.2 Image dimension invariant

Scope: To be image dimension invariant is for our system to have the ability to run with any sized image so that it is robust and general enough to tackle a multitude of different medical scenarios.

Computing Environment Minimum Requirements:

RAM: Minimum 4GB Ram

Hard Drive Space: 1.4GB

2GHz processor

Screen Resolution: 1920x1080

Step by step:

1.) Process an image through Fractal Eyes system
2.) Record features extracted
3.) Downsample original image
4.) Reprocess the downsampled image through Fractal Eyes system
5.) Record features extracted and compare against previous set of features extracted to see whether results are consistent
6.) Repeat steps 3-5 a second time

5.2.2 Features of Interest Extraction Test Plan

Scope: The a-priori feature extraction routine will be tested to verify accuracy of feature extraction based on predetermined variable sets.

Computing Environment Minimum Requirements:

RAM: Minimum 4GB Ram

Hard Drive Space: 1.4GB

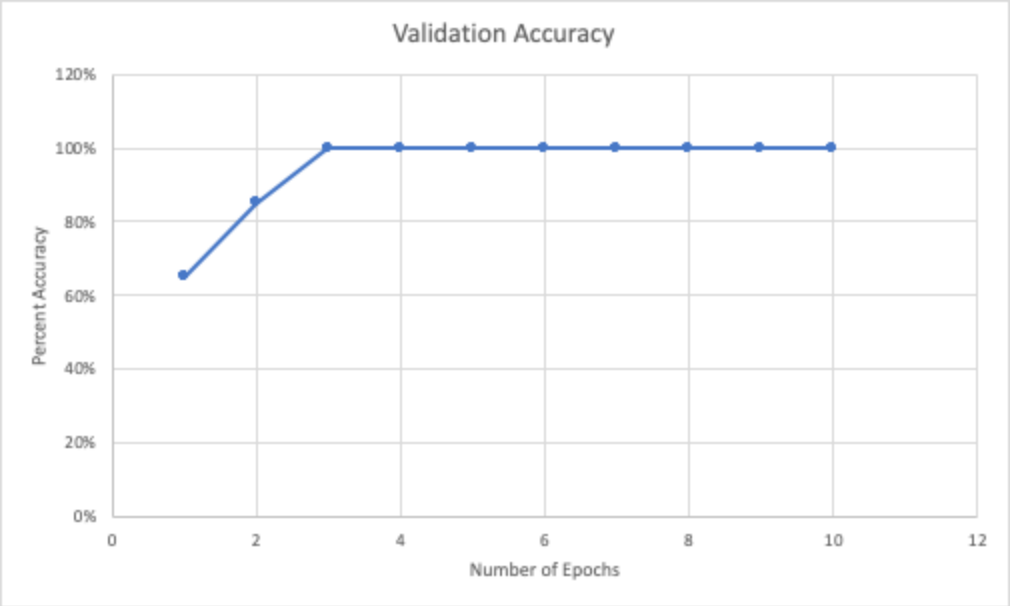
2GHz processor

Screen Resolution: 1920x1080

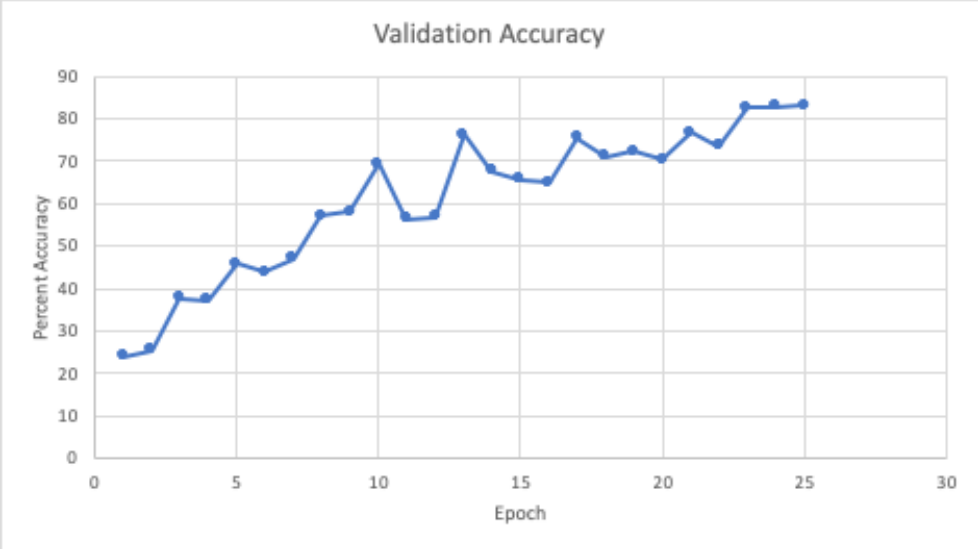
Step by step processed for this test vary depending on feature of interest. The features of interest are determined by type of image being examined (MRI, histological, etc) and the quality of data. The general process is to manually determine a quantifiable quality, develop the extraction process and compare the result of the extraction process to that of the manual calculation. See Section 5.2.2 “Features of Interest Extraction” on the Software Design Document for list of features extracted in this application of the project as well as the verification matrix for this subsystem.

Models and Analyses

This project did require any models. However, there are analyses of the data that are useful to discuss. Below are details of the analyses of our data.



Graph 1: Validation Accuracy of Neural Network Classification Subsystem



Graph 2: Validation Accuracy of Neural Network Classification Subsystem (Four Class Problem)

Graph 1 details the accuracy of the Neural Network subsystem used to classify the U2OS and White Blood Cell datasets. In the Predictions of Performance Margins and Accuracy we expected a validation accuracy of around 80% due to the relatively small size of our dataset. This prediction severely underestimated the power of Convolutional Neural Networks in image classification.

Graph 2 details the accuracy of the Neural Network subsystem used to classify different White Blood Cell classes. The accuracy here is not quite as high as in the two classes problem as 1) this implementation required to differentiate between an additional two classes than the former and 2) the four white blood cell classes are much similar to one another than the U2OS cell was to White Blood Cells. Despite this, an accuracy of 82.7% was achieved still beating our prediction of 80%.

Table 1: Mutual Information Scores: Feature vs Class (2-Class)

# of Blobs[1]	Average Area	Average Red	Average Blue	Average Green	Average Luminosity
0.63490236	0.69383258	0.69383258	0.69383258	0.69383258	0.69383258

Table 2: Mutual Information Scores: Feature vs Feature (2-Class)

	# of Blobs[1]	Average Area	Average Red	Average Blue	Average Green	Average Luminosity
# of Blobs[1]	4.8700	1.2600	0.0866	0.0907	0.0702	0.0631
Average Area	1.2600	6.3100	0.0105	0.0478	0.0262	0.0452
Average Red	0.0899	0.0105	6.3100	1.4500	1.3700	1.6800
Average Blue	0.0903	0.0477	1.4500	6.3100	1.7100	2.6800
Average Green	0.0713	0.0262	1.3700	1.7100	6.3100	1.8800
Average Luminosity	0.0636	0.0452	1.6800	2.6800	1.8800	6.3100

Above are tables detailing the mutual information scores for the initial two class differentiation.

Table 1 shows that when differentiating between U2OS cells and White Blood Cells all of the features examined are roughly equal differentiators (as they have roughly the same mutual information score). It could be argued that number of blobs is a slightly less useful differentiator given that it has the lowest mutual information score. However, more research is needed to determine if the difference in that value from the others is statistically significant. This is expanded upon in Recommendations For Future Work in the Appendix.

Table 2 details the mutual information score between features. That is, how closely is one feature related to another feature. It should be noted that 1) the scores along the diagonal of the

[1]Blobs = features extracted by Difference of Gaussian, Laplacian of Gaussian, and Determinant of Hessian

table should be ignored as that is the mutual information score of a feature with itself which would logically be high and 2) this table is symmetric so reading the top diagonal of the table will give you the same information as reading the bottom diagonal.

This table shows that the color channels are all very related to one another and that average area of a blob is closely related to the number of blobs.

Table 3: Mutual Information Scores: Feature vs Class (4-Class)					
# of Blobs[1]	Average Area	Average Red	Average Blue	Average Green	Average Luminosity
0.62900	0.5770	0.00932	0.0337	0.0266	0.0393

Table 4: Mutual Information Scores: Feature vs Feature (4-Class)						
	# of Blobs[1]	Average Area	Average Red	Average Blue	Average Green	Average Luminosity
# Blobs[1]	4.8700	1.2600	0.0874	0.0922	0.0683	0.0640
Average Area	1.2600	6.3100	0.0105	0.0477	0.0262	0.0452
Average Red	0.0900	0.0105	6.3100	1.4500	1.3700	1.6800
Average Blue	0.0902	0.0477	1.4500	6.3100	1.7100	2.6800
Average Green	0.0675	0.0262	1.3700	1.7100	6.3100	1.8800
Average Luminosity	0.0649	0.0452	1.6800	2.6800	1.8800	6.3100

Table 3 shows that when differentiating between the four classes of white blood cells, color is much less useful than the number of blobs and the average area of those blobs. Exactly whether or not these numbers mean that color is “not useful” in differentiating white blood cells remains to be determined. This is explore more in Recommendation For Future Work in the Appendix.

Table 4 follows the same rules outlined in Table 2. There is not more that needs to be expanded upon in this section.

[1]Blobs = features extracted by Difference of Gaussian, Laplacian of Gaussian, and Determinant of Hessian

Acceptance Test Results

<table>
<tr><th colspan="4">Acceptance Test Data Sheet</th></tr>
<tr><td colspan="4">Referenced ATP Paragraph Number: 5.1.2 Image Dimension Invariant</td></tr>
<tr><td colspan="4">Analysis Referenced (for verification by T/A): Invariance to image dimensionality producing analogous feature values across multiple image sizes</td></tr>
<tr><td colspan="4">Name of Test: Image Dimension Invariant</td></tr>
<tr><td colspan="4">Unit Under Test (UUT): System
Name: Fractal Eyes
Part Number: n/a
Serial Number: n/a</td></tr>
<tr><td colspan="2">Results (Pass / Fail): Pass[1]</td><td colspan="2">Date of Test: 4/2/19</td></tr>
<tr><td>Recording of Test Measurement:</td><td>Requirement (SRD, with Tolerances): 5.1.2 Image Dimension Invariance</td><td>Test Equipment Error: n/a</td><td>Adjusted Test Limit:</td></tr>
</table>

Computations, (Include Analyses Results, if any):

Average Values of Features Per Downsample Size						
Dimensions/ Feature	Average Red	Average Green	Average Blue	Average Luminosity	Average Blobs	Average Area
200px x 200px	193.10722	189.19595	198.46845	190.69696	66	323.00028
100px x 100px	194.7629	190.8392	200.0416	192.33779	31	221.23759
50px x 50px	200.4004	196.4872	206.1876	198.01951	8	323.46328

Signatures:

Tester: Andrew Burger

Customer: Dr. Marvin J. Slepian

[1]While results of Average Blobs and Average Area of blobs changed across different image dimensions, results are consistent within a downsample interval. That is, on average images with dimensionality 100px x 100px will have around 30 blobs for this dataset. Thus when training the system. It is recommended to have all images with the same dimensionality in order to achieve consistent and meaningful results. This idea is expanded on in Recommendations For Further Work below.

<table>
<tr><th colspan="4">Acceptance Test Data Sheet</th></tr>
<tr><td colspan="4">Referenced ATP Paragraph Number: 5.2.2</td></tr>
<tr><td colspan="4">Analysis Referenced (for verification by T/A):</td></tr>
<tr><td colspan="4">Name of Test: Features of Interest Extraction</td></tr>
<tr><td colspan="4">Unit Under Test (UUT): Features of Interest Extraction Subsystem
Name: FEI
Part Number: n/a
Serial Number: n/a</td></tr>
<tr><td colspan="2">Results (Pass / Fail): Pass</td><td colspan="2">Date of Test: 2/13/19</td></tr>
<tr><td>Recording of Test Measurement: See Section 5.2.2 Features of Interest Extraction in TDP for results</td><td>Requirement (SRD, with Tolerances): 5.2.2 Features Of Interest Extraction</td><td>Test Equipment Error: n/a</td><td>Adjusted Test Limit: n/a</td></tr>
</table>

Computations, (Include Analyses Results, if any):

See Section 5.2.2 Features of Interest Extraction in TDP for results

Signatures:

Tester: Diego Alcantara

Customer: Dr. Marvin J. Slepian

Final Budget

Item	Unit Cost	Quantity	Cost
Custom Embroidered Polos	$22.27	5	$111.35
Poster 36" x 48" w/ printing	$141.30	1	$141.30
		Total Cost	$252.65
		Remaining Budget	$3,747.35

Lessons Learned

Developing a program over an extended period of time can be difficult to manage. The project deadline can feel far off into the future an may incentivize developers to not spend much time on it early on. In addition, having a broad set of requirements can make it seem like there is no easy place to start. Because of this, we as a team, learned the value of breaking up the project into smaller parts and setting close deadlines to force ourselves to make progress. This method was invaluable in staying on top of the project and completing it in a timely manner to allow ourselves to handle not only development, but the logistical requirements of Senior Design as well.

Furthermore, the intrinsic structure of the ENGR498 course was fundamental to the development of our technology. It became increasingly apparent to us as group that the hard deadlines implemented and distributed throughout two semesters helped guide the conception and implementation of the project. Thus from an engineering point of view, the value of project management and goal oriented approaches to development became increasingly apparent.

Interestingly, there are several social learning outcomes that, we as a group, came to experience. One of them was the subtle art of communication. Over the course of two semesters, the manner in which we communicated with one another, specifically when pertaining to the project, gradually changed, in a sense experiencing a progressive evolution. Because we were dealing with an interdisciplinary project it became apparent that individuals within the group had particular strengths and weaknesses. The respective strengths of the the individuals within the group became pronounced as we progressed with the development of our project; where, tasks began to be

distributed based on the relative familiarity a team member had with the task. Thus, assigning and distributing tasks became increasingly effortless.

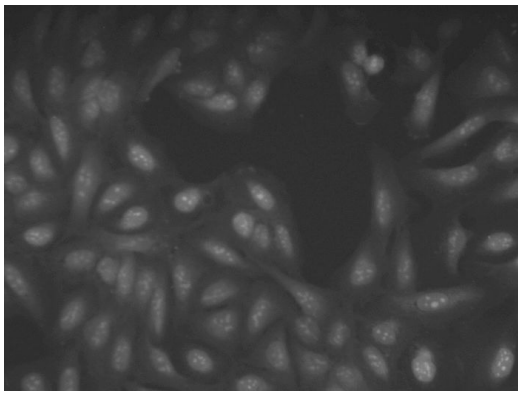
Building on the notion of communication, a lesson that we cherished as a group was the unexpected degree of consideration we developed as a group. Often times, individuals had tough weeks/schedules and struggled to meet certain deadlines. Yet, because we valued the overall success of the team, and we had established a goal oriented approach, we developed an unspoken level of trust amongst one another when concerning hard deadlines. We communicated well with one another and developed discussing problems we were having with certain tasks and asking for help when needed. This inevitably helped increase our productivity and decrease the overall stress within the group. The task distribution approach we implemented fit well with our intra-personal skills and further helped develop a sense of cohesion with the group as we progressed with the project. Thus, over the course of 32 weeks we inevitably developed a lifelong skill: how to communicate effectively and truthfully.

The interdisciplinary nature of the project also taught us a sense of value and utility for the skills we had accrued throughout our college careers. Bi-weekly discussions or “updates” as we coined them, were important for two reasons: namely, they enabled us to share our progress pertaining to our current tasks and develop a strategy to move forward with but they also helped us individually affirm the level of familiarity and confidence we had with a certain skill or aspect of the project. In sharing with one another the complexities of our portion of the project we inevitably learned from one another; holistically enriching the groups understanding of the technology we were aiming to develop as well as imparting some, at minimum, high level understanding of what each engineering field was contributing to its development. Thus, given enough time we began to have a more holistic view of the project and express, with a higher degree of confidence, the presumed utility of the technology we were building.

Appendix

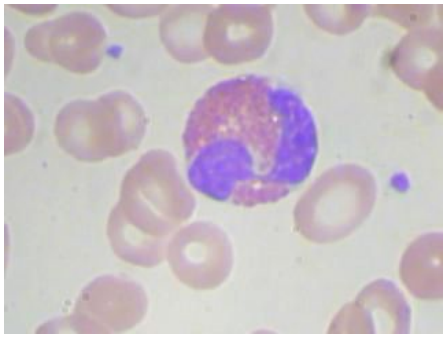
Samples From Datasets

Single Class U2OS Cells

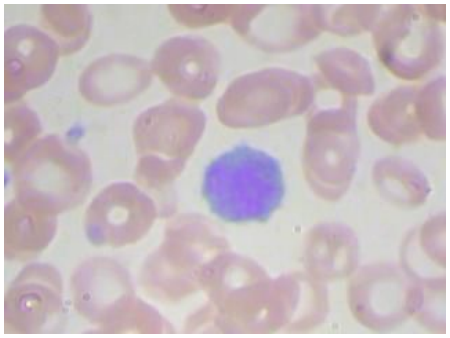


Picture 1: Sample of U2OS Cells

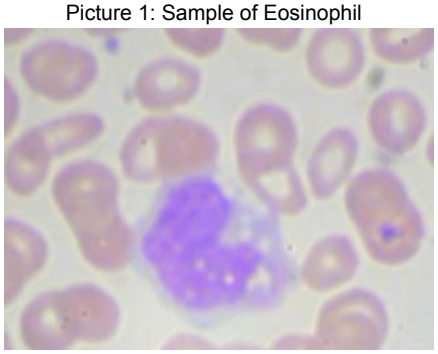
Four Classes of White Blood Cells



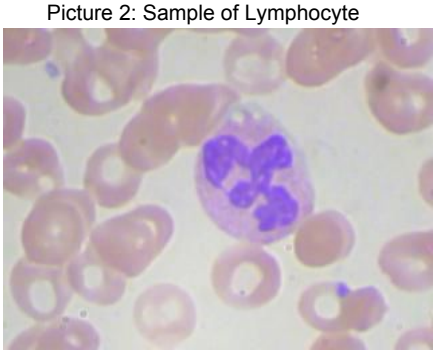
Picture 1: Sample of Eosinophil



Picture 2: Sample of Lymphocyte



Picture 3: Sample of Monocyte



Picture 4: Sample of Neutrophil

Operation of GUI

A video detailing the operation of the GUI can be found at: https://bit.ly/2XTT6Pk

Recommendations For Future Work

While major advances have been made in the development of this project, this iteration of the project is just a proof of concept. Further work and research is necessary. Below we list the major focuses of improving on this project.

1. Collect more features

The features used for analysis in this iteration of the project are somewhat limited in their use. While the features used for this iteration of the project were useful in determining differences between Human U2OS cells and Human White Blood cells, these features were not as useful determining differences between classes of white blood cells (an experiment we did to in an effort to explore how robust our program was).

One possible additional feature would be to look at the distribution of color within an image to search for higher density areas and then investigate differences in those high density areas as a determining feature.

We would recommend that the sponsor provide an expert in the field of histology (of whichever field this technology will be applied to next iteration) that can advise the students continuing this work on what features are valuable and important so that the students can develop a feature extraction routine around said advice. The technology can also be used to find a quantitative basis for determining the value of features that experts are still unsure of.

The more features used for analysis, the more valuable the results can become.

2. Add support for automatic downsampling

As stated in Image Dimension Invariance Test Data Sheet above, in order to have consistent, meaningful results across a dataset, images must be of similar dimensionality. Using an image dataset with some images of dimensionality 512px x 512px and other images of dimensionality 396px x 396px may lead to ambiguous and inconsistent results. Images with a higher dimensionality have higher resolution and thus more pixel information. It it therefore easier for the "blob" (Difference of Gaussian, Laplacian of Gaussian, and Determinant of Hessian) features to be determined for higher resolution images. More blobs may appear for higher resolution images than for lower resolution images. Thus, downsampling to the smallest image within one's dataset is highly recommended.

Within medical imaging, there are many constraints put on images that will be used in a dataset. Having consistent dimensionality may not be an issue for most datasets, but it would be a useful feature to have in the event some images are of different dimensionalities within a dataset.

It should be noted that downsampling an image does reduce the amount of pixel information available for analysis. How this affects the end result and the value of the system is undetermined at this time and needs further research.

3. Expand on Natural Language Explanation

The current implementation of the natural language explanation simply displays the n most relevant features ordered by mutual information score as related to a class label and the n most relevant pairs of features ordered by mutual information score as related to each other. This implementation is rudimentary and does not explain whether the feature and its corresponding mutual information score are statistically valid differentiators or not.

It would be useful to determine what a “relevant” feature is based off of the numbers. In the two class implementation of this problem, the most relevant features had mutual information scores around 0.65 as it relates to a class label. In the four class implementation, the most relevant features had mutual information scores around 0.5-0.6 as it relates to a class label. Obviously a lower mutual information score implies less relation between that feature and a class, but at what threshold does the feature become “not relevant?” The two implementation we tested in this iteration are not enough to make any claims on this as it relates to histological imaging. Thus, further research is needed on a variety of datasets as well as this dataset.

4. Develop a GUI using a more robust GUI framework

The GUI developed in this iteration of the project is functional and usable albeit rudimentary. That being said, if the sponsor’s end goal is to create a marketable product, there are superior GUI building frameworks that should be used instead. Finding someone with expertise in Human Computer Interaction and User Interface/Experience Design would also be of value in creating a better end product.

In addition, to take full advantage of the program it would make sense add a window to “train” the program. That is, to train the neural network on a binary dataset of many images as well as taking advantage of the full_image_set_analysis function within the code that generates the csv from which all individually analyzed images will compared.

Another minor, but useful fix would be to add a field where the user can decide how many features and how many feature pairs they would like to see in the final report. Currently, the only way to modify this functionality would be to go into the source code and edit it directly. This would require a user to have at least a basic understanding of both programming and the code base.

A lot of the GUI refinement recommendations essentially come down to user experience modifications.