CS 567
Final Project
By Utkarsh Jain

Preprocessing

I used various steps for preprocessing the images.

1. Use only the green channel - I used only the green channel for my processing since it had the greatest contrast between the spots and the rest of the image.

2. Subtract median filtered image from original image - To remove the intensity gradient, I used a low pass filter (Median filter) to get an estimate of the low frequencies in the image. Then I subtracted the median filtered image from the original image which reduced the intensity gradient. The size of the kernel I used was 5*5 because a smaller filter didn't remove the intensity gradient and bigger median filters blurred the image too much and too much information was lost.

3. Windowing Function - Since my objective was to pull out the spots in the image and focus only on the higher intensities, I used a windowing function to make the higher intensities in the image more prominent. For each image, I used 0.61*(Max value in the image) for the lower limit of the windowing function and 0.95*(Max value in the image) as the upper limit of the windowing function. I used these bounds because after a lot of trial and error, these bounds gave me the best results for most of the images and made the spots more prominent.

4. Binarize - I used a threshold of 0.61*(Max value in the image) to binarize the image. I used this threshold because it made most of the background pixels 0 and most of the pixels in the spots 1.

5. Dilate - After performing the previous four operations, I noticed that some of the healthy images had multiple little spots that were close by while the unhealthy images had spots that were spread out. Since I knew that the close by spots in the healthy images were not important to me and the spots that I wanted the be counted as distinct were not close together, I dilated the image using imdilate. This merged the spots that were close by so there was a better distinction between the healthy and unhealthy images. I created a flat disk-shaped structuring element with a radius of 20 since the spots were round and the radius of 20 merged only the close by spots and not spots that were further apart. To create the structuring element, I used the strel function and then used the structuring element in the imdilate function.

6. Mask - After performing the above 5 steps, some images still had pixels that came from the outline of the eye and should have been classified as the background. To get rid of the outline, I made a mask of the eye. To make the eye of the mask, I thresholded the green channel of the image at 0.1*(Max value in the image) as it gave me the best mask of the eye in the image. Since I didn't want the outline of the eye in my processed images and knew that most of the spots were towards the inside of the eye, I eroded the mask with disk shaped structuring element of radius 60 and eroded

the eye, again using imerode. I used a radius of 60 because it gave me a mask that was sufficiently eroded.

Features

1. Number of Spots - After performing the preprocessing steps, I counted the number of distinct spots (connected components) in the image using bwconncomp. This function counts the number of connected components in the image similar to what we learned in class. After counting the number of spots in the image, I thresholded the value of the features so if the number of spots was greater than 1 then the feature value assigned to the image was 1 while if it was less than or equal to one, it was given a value of 0. I did this because I noticed that the healthy images usually had 1 to less spots while the unhealthy images had more than one spot.

2. Area - For my second feature, I counted the number of pixels with that value of 1 in the processed images because I saw that the unhealthy images had a larger number of pixels with the value of 1 as compared to the healthy images. For this feature, I simply took the sum of all the values in the image. I thresholded the value of the features so if the area was greater than 2500 then the feature value assigned to the image was 1 while if it was less than or equal to 2500, it was given a value of 0. I did this because I noticed that the healthy images usually had and area of less than or equal to 2500 and unhealthy images usually had areas that exceeded 2500.

Cross Validation

Scaling - I did not scale my features as both features had either a value of 1 or 0 and scaling would not help in this case. The code for how I would do the scaling is in the code but it is commented out.

1. K- Nearest Neighbors - The first classifier I used was the KNN classifier. The algorithm found the closest 7 points in the feature space of points not in the fold for each of the points in the fold. It assigned the label of the majority of these 7 points as the label of the test point. I found the value of k using trial and error and k=7 gave me the highest accuracy and thus I used it. The number of folds I used for KNN is 6 because using a low number of folds didn't give good accuracies and 6 folds gave me a reasonable accuracy.

2. Logistic Regression - The second classifier I used was Logistic Regression. For this I used glmfit function with a binomial distribution, link function, and logit was the link parameter. The glmfit function used the test points and gave me the coefficients as it's output. I multiplied these coefficients with corresponding feature values and put the value obtained in the sigmoid function to obtain a probability. If the probability was higher than 0.5 than the point was given the label of 1 (unhealthy) and if lower than 0.5 then the image was classified as healthy. For Logistic Regression, I again used 6 folds as 6 folds gave me the best results.