# HOW TO INCREASE THE ACCURACY OF GLAUCOMA DATASET USING HARALICK TEXT FEATURE GLCM TECHNIQUE ON MATLAB SOTFWARE

From the previous project it as obvious the accuracy we got after training was not as high as we expected. Comments and suggestion from fellow researcher helped me realized more work can be done to help me improve on my algorithm and accuracy.

After much consultation I realized some of the best way to increase on once accuracy are:

a) Adding more data.
b) Increasing the number of features.
c) Redesigning the neural network architecture ( adding more neuron and hidden layers)

   Note: this depends on the type of dataset.
d) Increasing the number of iteration or epoch.
e) Increasing the time we train and retrain the model.
f) Trying different classification model and comparing.

Demerit:

One of the greatest challenged faced during this research was the unavailability of data. So this made it difficult to get a perfect system model.

Previously I used 5 different features for building up my model which did not have much accuracy.

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | CONTRAS | CORRELAT | ENERGY | Homogen | ENTROPY | OUTPUT |
| 2 | 0.0944 | 0.9671 | 0.2584 | 0.9546 | 1.6474 | 1 |
| 3 | 0.0817 | 0.9636 | 0.2808 | 0.9593 | 1.6774 | 1 |
| 4 | 0.0755 | 0.9685 | 0.2499 | 0.9628 | 1.688 | 1 |
| 5 | 0.0811 | 0.9684 | 0.2312 | 0.9598 | 1.7332 | 1 |
| 6 | 0.052 | 0.9772 | 0.286 | 0.9744 | 1.4939 | 1 |
| 7 | 0.0687 | 0.97 | 0.2936 | 0.9661 | 1.4788 | 1 |
| 8 | 0.0609 | 0.974 | 0.2934 | 0.9706 | 1.4974 | 1 |
| 9 | 0.0656 | 0.9744 | 0.3106 | 0.9682 | 1.479 | 1 |
| 10 | 0.0739 | 0.9717 | 0.3348 | 0.9648 | 1.458 | 1 |
| 11 | 0.0515 | 0.9789 | 0.3349 | 0.9751 | 1.3912 | 1 |
| 12 | 0.0732 | 0.9656 | 0.2614 | 0.9638 | 1.5694 | 1 |
| 13 | 0.0625 | 0.9488 | 0.3821 | 0.9691 | 1.3299 | 1 |
| 14 | 0.0601 | 0.9538 | 0.3897 | 0.9711 | 1.2321 | 1 |
| 15 | 0.0748 | 0.9453 | 0.3513 | 0.9634 | 1.3265 | 1 |
| 16 | 0.0447 | 0.9729 | 0.3974 | 0.9797 | 1.2382 | 1 |
| 17 | 0.0382 | 0.9776 | 0.4027 | 0.9815 | 1.2102 | 1 |
| 18 | 0.0489 | 0.9676 | 0.3476 | 0.9759 | 1.3598 | 1 |
| 19 | 0.0555 | 0.9664 | 0.3128 | 0.9726 | 1.4373 | 1 |
| 20 | 0.0574 | 0.9636 | 0.3828 | 0.9726 | 1.2583 | 1 |
| 21 | 0.0685 | 0.969 | 0.2878 | 0.9666 | 1.5525 | 1 |
| 22 | 0.0686 | 0.9656 | 0.2977 | 0.9662 | 1.5542 | 1 |
| 23 | 0.0661 | 0.9608 | 0.3333 | 0.9674 | 1.419 | 1 |

Further research was made, and more application on the GLCM techniques was done.

**Feature Extraction** This process extracts the gray-tone spatial-dependence matrices. The GLCM is a technique that evaluates texture of image by taking into attention the spatial relationship of pixels.

The Haralick Texture Feature Extraction is used I getting the following features:

I. Contrast

$$\sum_i \sum_j |i-j|^2 p(i,j)$$

II. Correlation

$$\sum_i \sum_j \frac{(i-\mu_x)(j-\mu_y)p(i,j)}{\sigma_x \sigma_y}$$

III. Energy

$$\sum_i \sum_j p(i,j)^2$$

IV. Homogeneity

$$\sum_i \sum_j \frac{p(i,j)}{1+|i-j|}$$

V. ENTROPY

$$-\sum_i \sum_j p(i,j) \log_2(p(i,j))$$

VI. SUM OF SQUARE

$$\sum_i \sum_j (i-\mu)^2 p(i,j)$$

VII. INVERSE DIFFERENCE MOMENT

$$\sum_i \sum_j \frac{1}{1+(i-j)^2} p(i,j)$$

VIII. SUM AVERAGE

$$\sum_{i=2}^{2N_g} i\, p_{x+y}(i)$$

IX. SUM VARIANCE

$$\sum_{i=2}^{2N_g} (i-f_8)^2 p_{x+y}(i)$$

X. SUM ENTROPY

$$-\sum_{i=2}^{2N_g} p_{x+y}(i) \log\{p_{x+y}(i)\}$$

XI. DIFFERENCE VARIANCE

variance of $p_{x-y}$

## XII. DIFERENCE ENTROPY

$$\sum_{i=0}^{N_g-1} p_{x-y}(i)\log_2(p_{x-y}(i))$$

## XIII. INFORMATION OF CORRELATION 1

$$\frac{Entropy - H_{XY1}}{\max(H_X H_Y}$$

## XIV. INFORMATION OF CORRELATION 2

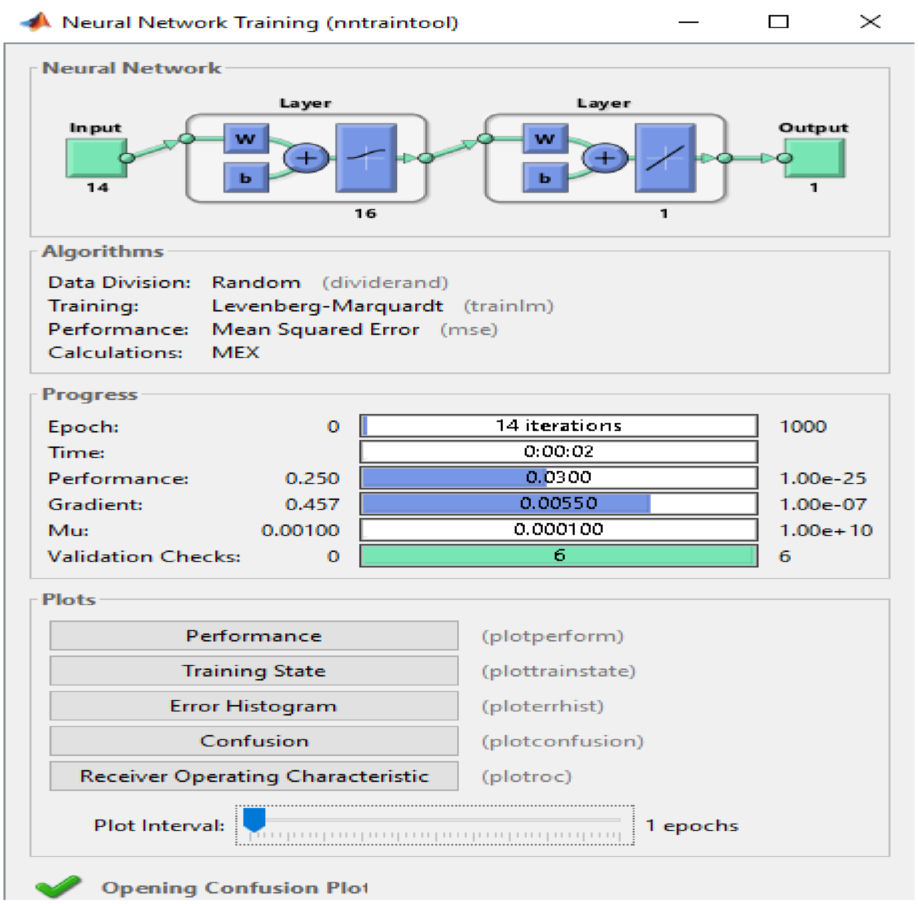
$$\sqrt{(1-\exp(-2(H_{XY2}-Entropy)))}$$

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | CONTRAST | CORRELAT | ENERGY | Homogen | ENTROPY | SUM OF S( | INVERSE D | SUM AVEF | SUM VARI | SUM ENTR | DIFFEREN( | D.ENTROF | INFO.COR | INFO.COR | OUTPUT |
| 2 | 0.0944 | 0.9671 | 0.2584 | 0.9546 | 1.6474 | 12.3634 | 0.9986 | 6.643 | 31.2838 | 1.5802 | 0.0944 | 0.3103 | -0.75 | 0.9282 | 1 |
| 3 | 0.0817 | 0.9636 | 0.2808 | 0.9593 | 1.6774 | 12.1385 | 0.9987 | 6.6695 | 29.8962 | 1.6203 | 0.0817 | 0.2829 | -0.7675 | 0.9361 | 1 |
| 4 | 0.0755 | 0.9685 | 0.2499 | 0.9628 | 1.688 | 9.6366 | 0.9988 | 5.8406 | 22.4126 | 1.6341 | 0.0755 | 0.2671 | -0.7923 | 0.9438 | 1 |
| 5 | 0.0811 | 0.9684 | 0.2312 | 0.9598 | 1.7332 | 10.1867 | 0.9988 | 5.9978 | 23.7406 | 1.6756 | 0.0811 | 0.2813 | -0.786 | 0.9455 | 1 |
| 6 | 0.052 | 0.9772 | 0.286 | 0.9744 | 1.4939 | 10.1525 | 0.9992 | 6.036 | 25.4828 | 1.4564 | 0.052 | 0.2035 | -0.8313 | 0.9384 | 1 |
| 7 | 0.0687 | 0.97 | 0.2936 | 0.9661 | 1.4788 | 10.5563 | 0.9989 | 6.1673 | 26.9519 | 1.43 | 0.0687 | 0.2498 | -0.7864 | 0.9235 | 1 |
| 8 | 0.0609 | 0.974 | 0.2934 | 0.9706 | 1.4974 | 10.578 | 0.9991 | 6.1655 | 26.8391 | 1.453 | 0.0609 | 0.2271 | -0.8096 | 0.9325 | 1 |
| 9 | 0.0656 | 0.9744 | 0.3106 | 0.9682 | 1.479 | 11.6204 | 0.999 | 6.4621 | 30.3562 | 1.4322 | 0.0656 | 0.24 | -0.7953 | 0.9264 | 1 |
| 10 | 0.0739 | 0.9717 | 0.3348 | 0.9648 | 1.458 | 12.3194 | 0.9989 | 6.6685 | 32.8435 | 1.4055 | 0.0739 | 0.2596 | -0.7719 | 0.9166 | 1 |
| 11 | 0.0515 | 0.9789 | 0.3349 | 0.9751 | 1.3912 | 11.3317 | 0.9992 | 6.3909 | 30.2028 | 1.3538 | 0.0515 | 0.2006 | -0.8252 | 0.9265 | 1 |
| 12 | 0.0732 | 0.9656 | 0.2614 | 0.9638 | 1.5694 | 9.3897 | 0.9989 | 5.8032 | 22.548 | 1.5172 | 0.0732 | 0.2615 | -0.7861 | 0.9322 | 1 |
| 13 | 0.0625 | 0.9488 | 0.3821 | 0.9691 | 1.3299 | 9.9401 | 0.999 | 6.1416 | 25.9614 | 1.2854 | 0.0625 | 0.2332 | -0.7665 | 0.8993 | 1 |
| 14 | 0.0601 | 0.9538 | 0.3897 | 0.9711 | 1.2321 | 10.8401 | 0.9991 | 6.4168 | 29.8781 | 1.1884 | 0.0601 | 0.2244 | -0.7679 | 0.8859 | 1 |
| 15 | 0.0748 | 0.9453 | 0.3513 | 0.9634 | 1.3265 | 10.8099 | 0.9989 | 6.3978 | 28.9219 | 1.273 | 0.0748 | 0.2649 | -0.736 | 0.887 | 1 |

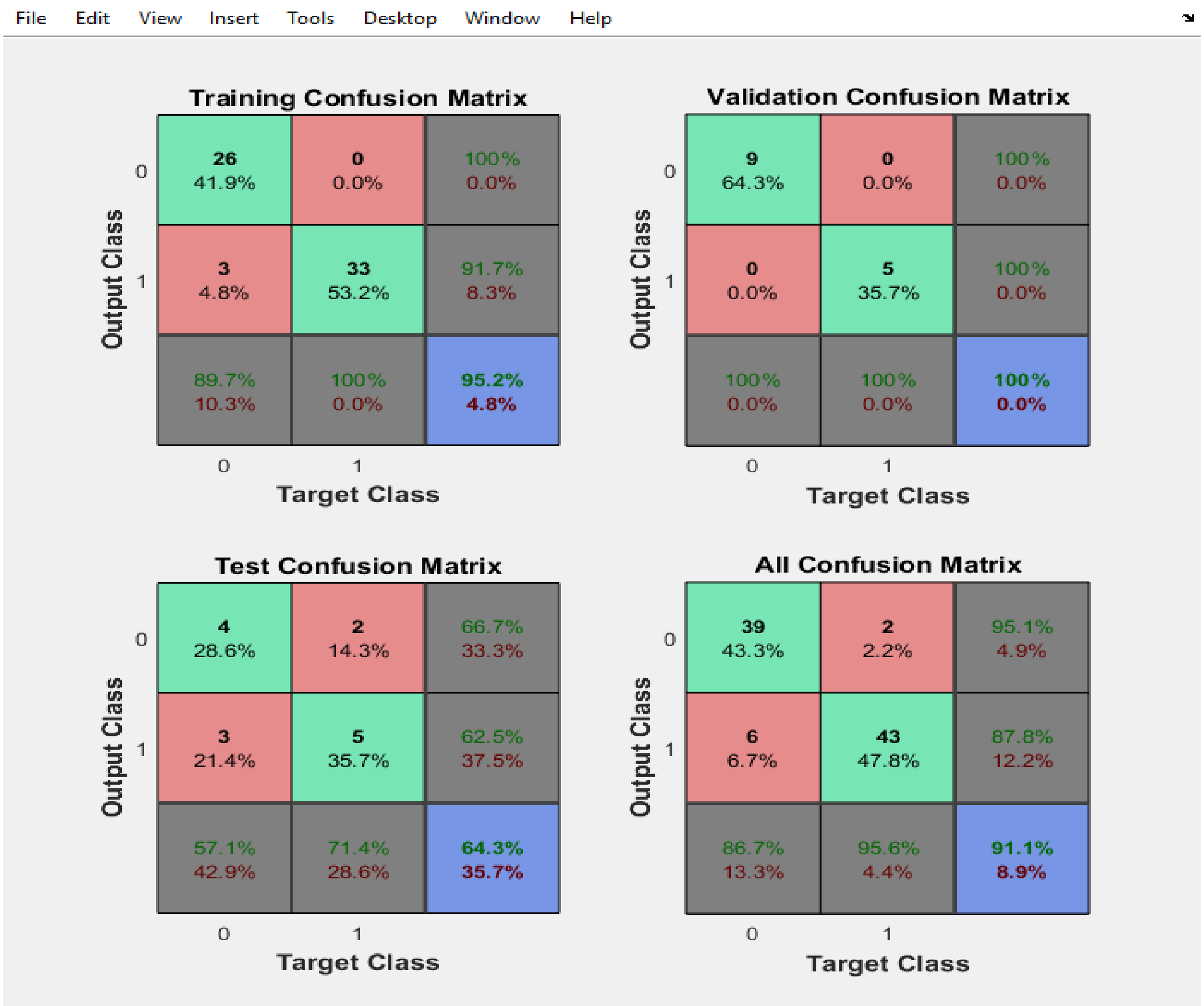
With the further feature extraction it can be seen that the number of feature increases.

## RESULTS

Firstly we have to train the neural network




From the training of the network we can get the confusion matrix, regression and performance of the network created.




The confusion matrix is a mathematical formula that helps show how accurate an algorithm or model can be.

We can see that the accuracy have improved compared to the previous accuracy 84.3% to 91.1%.

**PREDICTION**

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.07 | 0.9676 | 0.2761 | 0.9657 | 1.6461 | 8.9773 | 0.9989 | 5.6548 | 20.7219 | 1.5955 | 0.07 | 0.2526 | -0.7953 | 0.9416 | 1 |
| 0.0609 | 0.9389 | 0.3447 | 0.97 | 1.2671 | 5.9522 | 0.9991 | 4.7027 | 14.0424 | 1.223 | 0.0609 | 0.2288 | -0.771 | 0.8922 | 1 |
| 0.07722 | 0.962 | 0.2792 | 0.9639 | 1.6119 | 7.4606 | 0.9989 | 5.1365 | 16.4993 | 1.5619 | 0.0722 | 0.2592 | -0.7852 | 0.9357 | 1 |
| 0.0477 | 0.9379 | 0.4366 | 0.9762 | 1.1256 | 3.7831 | 0.9993 | 3.7187 | 8.3871 | 1.0924 | 0.0477 | 0.1918 | -0.7752 | 0.8715 | 1 |
| 0.0606 | 0.9119 | 0.479 | 0.9697 | 1.0866 | 3.7674 | 0.9991 | 3.7309 | 8.5309 | 1.0446 | 0.0606 | 0.2286 | -0.711 | 0.8359 | 1 |
| 0.0743 | 0.9336 | 0.3522 | 0.964 | 1.3376 | 5.2736 | 0.9989 | 4.3735 | 11.7203 | 1.2818 | 0.0743 | 0.2627 | -0.755 | 0.896 | 0 |
| 0.1179 | 0.894 | 0.3301 | 0.9543 | 1.3577 | 6.2573 | 0.9983 | 4.8047 | 14.4778 | 1.2876 | 0.1179 | 0.3129 | -0.6938 | 0.8742 | 0 |
| 0.0837 | 0.9141 | 0.4155 | 0.9582 | 1.2667 | 4.8629 | 0.9987 | 4.2153 | 10.9039 | 1.2086 | 0.0837 | 0.2877 | -0.709 | 0.8668 | 0 |
| 0.093 | 0.913 | 0.4932 | 0.9671 | 1.1011 | 7.3228 | 0.9987 | 5.2402 | 19.5998 | 1.0503 | 0.093 | 0.2483 | -0.7181 | 0.8423 | 0 |
| 0.1127 | 0.9427 | 0.2989 | 0.606 | 1.5144 | 10.3596 | 0.9984 | 6.1525 | 25.9657 | 1.4467 | 0.1127 | 0.2899 | -0.7512 | 0.9158 | 0 |
| 0.1179 | 0.894 | 0.3301 | 0.9543 | 1.3577 | 6.2573 | 0.9983 | 4.8047 | 14.4778 | 1.2876 | 0.1179 | 0.3129 | -0.6938 | 0.8742 | 0 |

Command Window

```
0.1179  0.894   0.3301  0.9543  1.3577  6.2573  0.9983  4.8047  14.4778 1.2876  0.1179  0.3129  -0.6938 0.8742
]';
>> y = net(testsample);
>> y

y =

    0.9831    1.1497    1.0102    0.9944    1.5650    0.4550    0.0587    0.2387   -0.0164    0.0054    0.0587

>> 
```