Disease Detection in Pomegranate using Image Processing

Rahul Waiker, Tushar Chavan, Prajyot Chatur, Aniket Chaudhari, Swanand Deshmukh

Mechanical Department- Vishwakarma Institute of Technology, Pune

***Abstract* — Manually, health monitoring and detection of disease in plants is difficult. Hence, image processing can be a useful and time saving tool for the detection of plant diseases. Diseases are classified based on the colour features and edge information. Images captured using mobile cameras are pre-processed, followed by segmentation, extraction of features and classification of diseases. Algorithms to detect the diseases will be developed on OpenCV platforms. This paper discussed the methods used in the detection of pomegranate diseases.**

***Keywords* — OpenCV, Grab Cut Segmentation, Feature extraction, GLCM.**

I. Introduction

Agriculture is the primary source of livelihood for large percentage of India's population, but contribution of agriculture to the national economy is less. It is due to lack of agriculture productivity. The productivity decreases because of plant diseases. So, it is necessary to identify the disease and getting solution for the same is very important. The solution mainly includes pesticides, weedicides and fertilizers to overcome the infection in scientific manner.

Fruits are very much prone to acquire the diseases very easily. Among the fruits, pomegranate is one of the fruits acquires the disease very easily and frequently. Pomegranate has high economic values and diseases in productivity affect the farmers adversely, so identification of diseases in pomegranate is very much necessary. It is used for medicinal purpose because of its high nutritional value. Pomegranate grows mainly in the area having annual rainfall around 40-50 cm. So, pomegranate is grown in arid and semi-arid regions. Pomegranates are used in medical field in curing diabetes, diarrhea and to overcome male and female infertility.

Nowadays farmers are facing many problems in cultivation of pomegranate mainly in the early stages. In early stages, the immunity of the pomegranate towards the infection is very less. Hence these fruits acquire the diseases very easily through infection causing pathogens.

In Agriculture, health monitoring of plants is really difficult to be done manually. Hence, automation is needed in the field of agriculture for disease detection, especially for fruit bearing plants. This project aims to solve the problem of health monitoring of pomegranate plants.

II. Literature Review

Our topic is quite related to Computer Vision and Image Processing and till now not enough work is done on this particular topic.

We have studied research papers thoroughly based on this project and we have drawn some insights from them regarding our topic.

We have also referred the course taken by ISRO and IIRS on Texture analysis which are on the same lines of our topic.

In the process, we have also referred the OpenCV-Python documentation for installation and practical coding of the project.

Yubing Li, Liangcheng Jiang, Peng Gao, Jinbo Zhang, Ming Chen [3] discussed an improved grab-cut segmentation algorithm where segmentation is done by a combination of graph cut and Grab Cut image segmentation. This method overcomes the drawbacks of simple grab-cut algorithm which cannot give accurate results when the background is a little similar to the interested area.

Zhao Xu, Wu Guoxin, Xu Baojie [4] proposed an OpenCV algorithm for Canny Edge Detection which depends on different threshold and same is convenient for the detection of edges in the image.

III. Methodology

A. Block Diagram

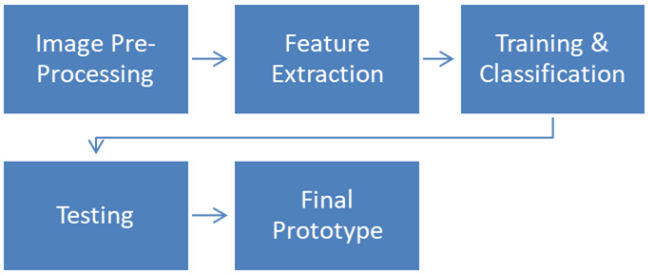


Fig.1: block diagram of process

B. Method

Image pre-processing:

1. Resizing: We maintain all images at particular size so as to simplify and reduce time taken to train the model. All image resized into 256X256 pixels.

2. Fourier Transform: As some images contains shine and it make difficult to segment. So that, Fourier Transform removes shine from image.

3. Grab-Cut Algorithm: This algorithm used to separate out foreground from its background.

Fig.2: a) Original fruit with disease b) Segmented image c) Fourier Transform d) Gray image

Feature extraction:

1. Canny Edge Detection

Canny Edge Detection is a popular edge detection algorithm. It was developed by John F. Canny in 1986. It is a multi-stage algorithm and we have to go through various stages.

1.1 Noise Reduction

Since edge detection is susceptible to noise in the image, first step is to remove the noise in the image with a 5x5 Gaussian filter.

1.2 Finding Intensity Gradient of the Image

Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get first derivative in horizontal direction (G_x) and vertical direction (G_y). From these two images, we can find edge gradient and direction for each pixel as follows:

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Fig.3: Formula for Edge Gradient and Angle

Gradient direction is always perpendicular to edges. It is rounded to one of four angles representing vertical, horizontal and two diagonal directions.

1.2 Non-maximum Suppression

After getting gradient magnitude and direction, a full scan of image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, pixel is checked if it is a local maximum in its neighborhood in the direction of gradient. Check the image below:

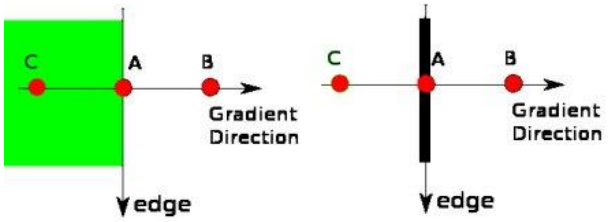


Fig.4: Edges and gradient directions

Point A is on the edge (in vertical direction). Gradient direction is normal to the edge. Point B and C are in gradient directions. So, point A is checked with point B and C to see if it forms a local maximum. If so, it is considered for next stage, otherwise, it is suppressed (put to zero). In short, the result you get is a binary image with "thin edges".

1.3 Hysteresis Thresholding

This stage decides which are all edges are really edges and which are not. For this, we need two threshold values, min_{Val} and max_{Val}. Any edges with intensity gradient more than max_{Val} are sure to be edges and those below min_{Val} are sure to be non-edges, so discarded. Those who lie between these two thresholds are classified edges or non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are considered to be part of edges. Otherwise, they are also discarded. See the image below:

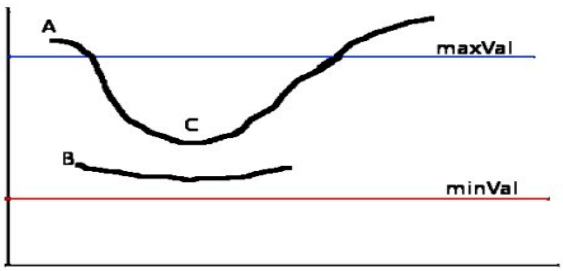


Fig.5: Range of values for intensity gradients

The edge A is above the max Val, so considered as "sure-edge". Although edge C is below max Val, it is connected to edge A, so that also considered as valid edge and we get that full curve. But edge B, although it is above min Val and is in same region as that of edge C, it is not connected to any "sure-edge", so that is discarded. So, it is very important that we have to select min Val and max Val accordingly to get the correct result. This stage also removes small pixels noises on the assumption that edges are long lines. So, what we finally get is strong edges in the image.

Fig.6: Output of canny edge algorithm

2. GLCM (grey level cooccurrence matrix)

A statistical method of examining texture that considers the spatial relationship of pixels is the gray-level co-occurrence matrix (GLCM), also known as the gray-level spatial dependence matrix. The GLCM functions characterize the texture of an image by calculating how often pairs of pixels with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix. (The texture filter functions, described in Calculate Statistical Measures of Texture cannot provide information about shape, that is, the spatial relationships of pixels in an image.)

After you create the GLCMs using Gray comatrix, you can derive several statistics from them using grayCoprs. These statistics provide information about the texture of an image.

Extracted features:

1. Contrast: It defers the calculation of the intensity contrast linking pixel and its neighbor over the whole image.

$$Contrast = \sum_{i,j=0}^{N-1} P_{ij}(i-j)^2$$

2. Dissimilarity: In homogenous image dissimilarity less while in heterogenous it will be more.

3. Homogeneity: Area which having same pixel value homogeneity is more

$$Homogeneity = \sum_{i,j=0}^{N-1} \frac{P_{ij}}{1+(i-j)^2}$$

4. Energy: Uniformity/repeating of pixel

$$Energy = \sum_{i,j=0}^{N-1} \left(P_{ij}\right)^2$$

5. Correlation: Measures the joint probability occurrence of the specified pixel pairs.

$$Correlation = \sum_{i,j=0}^{N-1} P_{ij} \frac{(i-\mu)(j-\mu)}{\sigma^2}$$

Training and classification:

Data set divided into training and testing part. 70% data use for training and remaining 30% use for testing/validation purpose. With the help of GLCM method, the features have been taken from the images one by one. All the data regarding the features of images have been collected by us in an Excel Sheet and the we have fed that data to the Random Forest Classifier. Total 353 images of Pomegranate consisting of different classes such as Class '0' is non-infected, Class 1 is Anthracnose, Class "2" is Alternaria Blackspot, Class :3" is Bacterial Plight, Class "4" Cercospora Fruit spot, Class "5" is Fruit rot, Class "6" is Fruit bore.

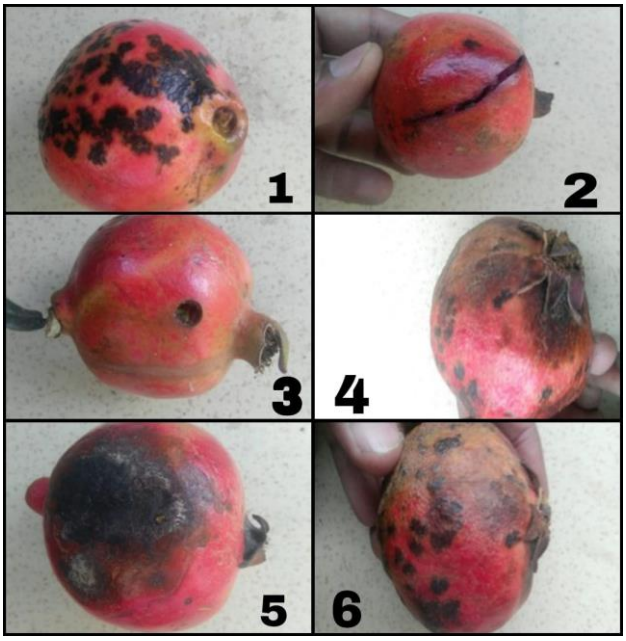
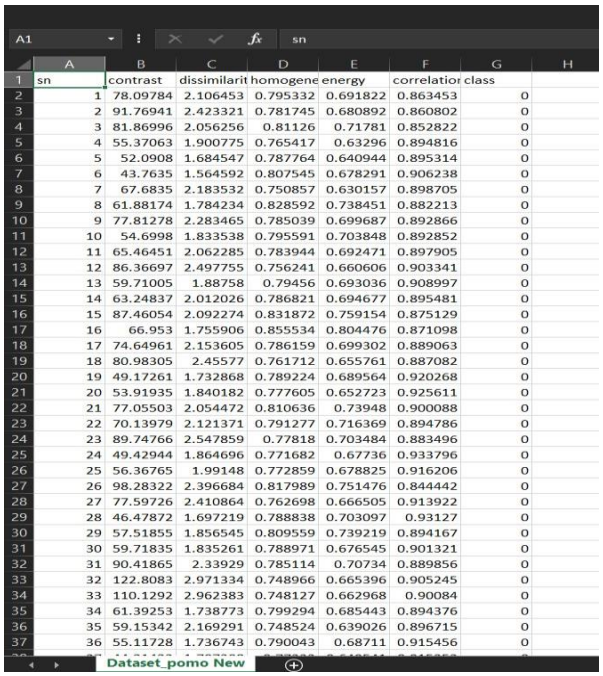


Fig.7: 1. Alternaria Blackspot 2. Bacterial Plight 3. Fruit bore 4. Cercospora Fruit spot 5. Fruit rot 6. Anthracnose

Dataset Review:



	A	B	C	D	E	F	G	H
1	sn	contrast	dissimilarit	homogene	energy	correlatio	class	
2	1	78.09784	2.106453	0.795332	0.691822	0.863453	0	
3	2	91.76941	2.423321	0.781745	0.680892	0.860802	0	
4	3	81.86996	2.056256	0.81126	0.71781	0.852822	0	
5	4	55.37063	1.900775	0.765417	0.63296	0.894816	0	
6	5	52.0908	1.684547	0.787764	0.640944	0.895314	0	
7	6	43.7635	1.564592	0.807545	0.678291	0.906238	0	
8	7	67.6835	2.183532	0.750857	0.630157	0.898705	0	
9	8	61.88174	1.784234	0.828592	0.738451	0.882213	0	
10	9	77.81278	2.283465	0.785039	0.699687	0.892866	0	
11	10	54.6998	1.833538	0.795591	0.703848	0.892852	0	
12	11	65.46451	2.062285	0.783944	0.692471	0.897905	0	
13	12	86.36697	2.497755	0.756241	0.660606	0.903341	0	
14	13	59.71005	1.88758	0.79456	0.693036	0.908997	0	
15	14	63.24837	2.012026	0.786821	0.694677	0.895481	0	
16	15	87.46054	2.092274	0.831872	0.759154	0.875129	0	
17	16	66.953	1.755906	0.855534	0.804476	0.871098	0	
18	17	74.64961	2.153605	0.786159	0.699302	0.889063	0	
19	18	80.98305	2.45577	0.761712	0.655761	0.887082	0	
20	19	49.17261	1.732868	0.789224	0.689564	0.920268	0	
21	20	53.91935	1.840182	0.777605	0.652723	0.925611	0	
22	21	77.05503	2.054472	0.810636	0.73948	0.900088	0	
23	22	70.13979	2.121371	0.791277	0.716369	0.894786	0	
24	23	89.74766	2.547859	0.77818	0.703484	0.883496	0	
25	24	49.42944	1.864696	0.771682	0.67736	0.933796	0	
26	25	56.36765	1.99148	0.772859	0.678825	0.916206	0	
27	26	98.28322	2.396684	0.817989	0.751476	0.844442	0	
28	27	77.59726	2.410864	0.762698	0.666505	0.913922	0	
29	28	46.47872	1.697219	0.788838	0.703097	0.93127	0	
30	29	57.51855	1.856545	0.809559	0.739219	0.894167	0	
31	30	59.71835	1.835261	0.788971	0.676545	0.901321	0	
32	31	90.41865	2.33929	0.785114	0.70734	0.889856	0	
33	32	122.8083	2.971334	0.748966	0.665396	0.905245	0	
34	33	110.1292	2.962383	0.748127	0.662968	0.90084	0	
35	34	61.39253	1.738773	0.799294	0.685443	0.894376	0	
36	35	59.15342	2.169291	0.748524	0.639026	0.896715	0	
37	36	55.11728	1.736743	0.790043	0.68711	0.915456	0	

Fig.8: Dataset containing all class features

Algorithm:

"Random Forest" algorithm used classification of images. Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithms of the same type i.e. multiple decision trees, resulting in a forest of trees, hence the name "Random Forest". The random forest algorithm can be used for both regression and classification tasks. The Random Forest Regressor class of the "Sklearn" library in Python is used to solve regression problems via Random forest.

The analogy and working of Random Forest algorithm is explained with the help of following diagram.

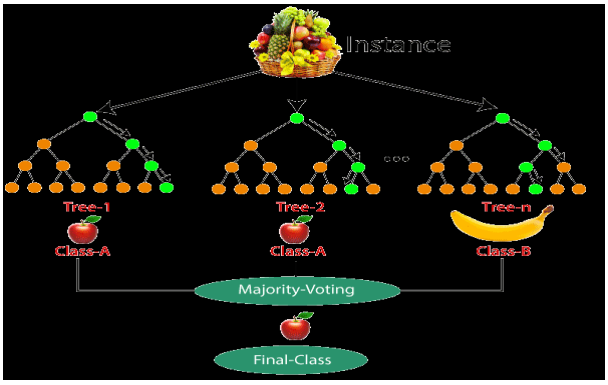


Fig.9: Decision Tree

Testing:

The model is tested on 151 images consisting of different classes. We have collected the features of testing images as well. On the basis of trained Random Forest Classifier, the model classifies the images according to the classes and gives the output.

IV. RESULTS AND DISCUSSIONS

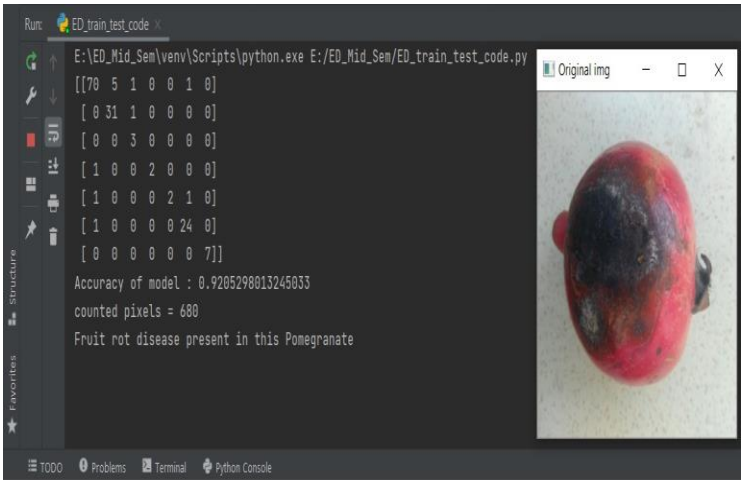


Fig.10: PyCharm terminal and bad pomegranate image

As we can see in the image above, we have to give input images to the model. The trained model then processes the image and classify it according to the class. The evaluation of models is done using certain parameters in Machine Learning. We have also used some parameters to evaluate our model. In Python Language, in Sklearn Library there is some intervention of evaluation parameters. We have used that parameters such as Confusion matrix to see how correct the model is predicting. Specifically, that is called as "Accuracy" in Machine Learning.

We have illustrated the methods below:

Confusion matrix:

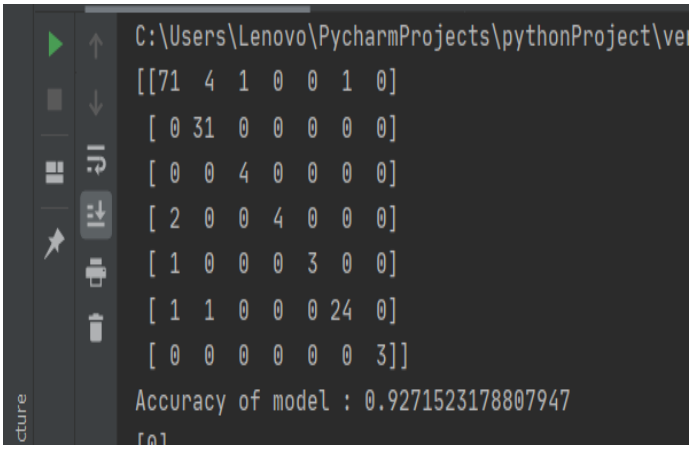


Fig.11: Confusion matrix

Confusion matrix is used to evaluate the models in Data Science. Above Confusion matrix gives the illustration of the model developed for classification of images according to the desired classes.

As we can see, the diagonal elements of matrix tell about the correct classified images according to the classes given in the format of testing images. Also, our model has achieved an accuracy of 92.71% that is also evaluated by the method of finding Confusion matrix in the Python language.

V. Limitations

This model is not deployed on any server or cloud Environment. Thus, the working may differ. Also, we have tested this model only on Standard Dataset. There might be a possibility that accuracy may vary according to the quality of captured image. Due to some problems, we were not able to go on real time farms and capture the real time images. One more Limitation is that, we have not developed any app or website as a frontend for this model. Thus, we can predict diseases only with help of Desktop or Laptop.

VI. Future Scope

Intent Search is the expected new module in the proposed system. Intent Search helps to capture the intension of the user while searching the information related to his fruit diseases. This module going to play major role when the quality of the input image is poor. In such case, the final result we are getting from the system may be poor. When user give his image input to his intent search module, it gives as output the number of images that are similar to the input image from the database. Then user has to click on the one image which he feels similar to his fruit diseases. That selected image by user is taken as input for the system. If the quality of input image is good in that case user may be skipping the Intent Search. Selection of Intent Search procedure is optional.

Also, the system can be deployed on an app or any Cloud Environment for the suitability of farmers so that they can use this system at any time irrespective of many gadgets and devices. They will just need a smartphone for using this system.

VII. Conclusion

On the basis of evaluation parameters given by the model, we can classify the defected images of Pomegranate according to the diseases such as Anthracnose, Alternaria Blackspot, Bacterial Plight, Cercospora Fruits pot, Fruit rot, Fruit bore etc. with an accuracy of 92.71% with the help of Random Forest Classifier. We have done this using a Standard dataset given by IEEE. With some optimization in this current model, we can deploy this model for real-time practices in Agriculture field.

VIII. References

[1] MANISHA bhangea, h.A. HINGOLIWALA “smart farming: pomegranate disease detection using image processing” second international symposium on computer vision and the internet (visionnet’15)

[2] Sharath D M, AKHILESH ROHAN M G, S Arun KUMAR, “Disease Detection in Pomegranate using Image Processing”, Proceedings of the Fourth International Conference on Trends in Electronics and Informatics (ICOEI 2020). IEEE Xplore Part Number: CFP20J32-ART; ISBN: 978-1-7281-5518-0

[3] Yubing LI, Jinbo ZHANG, Peng GAO, Liangcheng JIANG, Ming Chen, “Grab Cut Image Segmentation Based on Image Region”, IEEE 3rd International Conference on Image, Vision and Computing, PP.311-315, 2018.

[4] Zhao Xu, Xu Baojie, Wu Guoxin, “Canny edge detection based on Open 13th IEEE International Conference on Electronic Measurement & Instruments, PP.53-56, 2017.