Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 9, August 2021: 4740-4756

PREDICTION OF DISEASE & PESTICIDES USING IMAGE PROCESSING OF UNHEALTHY CROPS

Mr. Rohit Ghumare [a]

[a] Department of Computer Engineering (Mumbai University), Datta Meghe College of Engineering,

ghumare64@gmail.com

Mr. Abhishek Budar [b]

[b] Department of Computer Engineering (Mumbai University), Datta Meghe College of Engineering,

abhishekbudar28@gmail.com

Mr. Rohan Bordekar [c]

[c] Department of Computer Engineering (Mumbai University), Datta Meghe College of Engineering,

rohanbordekar@gmail.com

Prof. Nusrat Parveen [d]

[d] Department of Computer Engineering (Mumbai University), Datta Meghe College of Engineering,

np.cm.dmce@gmail.com

1. ABSTRACT

Agriculture is the mainstay of the Indian economy. Almost 70% of people depend on it and share a major part of the Gross Domestic Product (GDP). The economy of our country is heavily dependent on agriculture and if it is affected then it will have a major impact on the economy. Diseases happening on the crops are mainly on the leaves which affects the reduction of both quality and quantity of agricultural products. If the farmers completely rely on pure nakedeye observation of experts to detect and classify such diseases, it can prove to be very expensive and is not completely accurate. The improvement in the field of Artificial Intelligence can be proved useful towards finding a solution for diseases happening to crops. The study aims to design an application for automatic detection and classification of diseases happening to crops. Also, the application should provide fast, cheap, and accurate solutions for the task which can be of great realistic significance. The system will be built by integrating several intelligent technologies like Image processing and Convolutional Neural Network (CNN). The CNN model will be built using TensorFlow library Keras by defining convolutional layers which converts an image into 2D Convolve which further by using feature_extractor_layer gives 95% accuracy. The CNN model will be mainly focused on increasing the processing speed. Taking althese things into consideration we aim at building a system that identifies the disease that has affected the crop and predicts some pesticides to cure the disease.

2. INTRODUCTION

Detection of plant disease is a very vital study because, it could be beneficial in observing huge fields of crops and automatically detecting the disease based on the symptoms as soon as they develop on plant leaves.

In order to cultivate successfully, it is necessary to monitor crops for disease detection. Expert observation with the naked eye is the most common method used in practice. However, this necessitates expert monitoring on a continual basis, which could be prohibitively expensive in large farms.

Furthermore, in some developing countries, farmers may have to travel long distances to consult experts, making consulting experts both costly and time consuming. As a result, looking for a fast, automatic, less expensive, and accurate method to detect plant disease cases is critical.

Approximately 78% of farmers in the country are small and marginal, with limited resources. As a result, they are unable to use the optimal amount of inputs in their crops, which is critical for increasing productivity. Most farmers may not be aware of the pesticides required for crops, which may result in an unbalanced use of fertiliser, and they may also be unaware of which pesticide to use for the diseased crop. As a result, the yield suffers. Image processing is a method of performing operations on an image in order to improve it or extract useful information from it. The introduction of new technologies, such as digital image processing and image analysis, has numerous applications in the biological field. A user-friendly application installed on an Android phone may help farmers solve this problem. The farmer takes a picture of the leaf and then uploads it into the application. Further, the application identifies the disease by applying Image Processing techniques on the image. The details of the disease and the affected area along with the confidence, as well as the pesticide, are displayed to the farmer instantly. This could be useful for observing large fields of crops and automatically detecting disease with the help of symptoms as soon as they appear on leaves of the plant.

3. RELATED WORKS

Plant diseases have turned into a dilemma as it can cause significant reduction in both quality and quantity of agricultural products. Plant pests and diseases affect food crops, causing significant losses to farmers and threatening food security. The spread of transboundary plant pests and diseases has increased dramatically in recent years. Globalization, trade and climate change, as well as reduced resilience in production systems due to decades of agricultural intensification, have all played a part. Outbreaks and upsurges can cause huge losses to crops and pastures, threatening the livelihoods of vulnerable farmers and the food and nutrition security of millions at a time.

From this accumulated knowledge base, we can distil some general principles of plant disease control that can help us address the management of new problems on whatever crop in any environment. One such set of principles, first articulated by International Journal of Innovative and Emerging Research in Engineering (Volume 2, Issue 4, 2015), they concluded using only one algorithm namely K-means Clustering for Image processing techniques for several plant species that have been used for recognizing plant diseases. This reason limited the accuracy as there was no comparison in results of two or more algorithms [1].

The other research was carried out by Fr. Conceicao Rodrigues College of Engineering in which they proposed and evaluated a framework for detection and classification of plant leaf/stem diseases using image processing and neural network technique. A feed forward backpropagation neural network was configured and trained using extracted set of features and subsequently utilized for detection of leaf diseases. The drawback of this research was as only Neural Network was used and there was no comparison between the accuracies of several algorithm [2].

Kim et al. (2009) have classified the grape fruit peel diseases using color texture features analysis. The texture features are calculated from the Spatial Gray-level Dependence Matrices (SGDM) and the classification is done using squared distance technique. Grape fruit peel might be infected by several diseases like canker, copper burn, greasy spot, melanose and wind scar (Kim et al., 2009). Helly et al. (2003) developed a new method in which Hue Saturation Intensity (HIS) - transformation is applied to the input image, then it is segmented using Fuzzy C-mean algorithm. Feature extraction stage deals with the color, size and shape of the spot and finally classification is done using neural networks (Helly et al., 2003). Real time specific weed discrimination technique using multilevel wavelet decomposition was proposed by Siddiqil et al.(2009).In this histogram equalization is used for preprocessing. Features are extracted from wavelet decomposition and finally classified by Euclidean distance method (Siddiqil et.al, 2009) [3].

Al-Bashish et al. (2011) developed a fast and accurate method in which the leaf diseases are detected and classified using k-means based segmentation and neural networks based classification. Automatic classification of leaf diseases is done based on high resolution multispectral and stereo images (Bauer et al., 2011) [4]. Sugar beet leaves are used in this approach. Segmentation is the process that is carried out to extract the diseased region and the plant diseases are graded by calculating the quotient of disease spot and leaf areas. An optimal threshold value for segmentation can be obtained using weighted Parzen-window (Jun and Wang, 2008). This reduces the computational burden and storage requirements without degrading the final segmentation results [5].

Satish Kumar from Punjab University carried out a research on “Tomato Plant Disease Classification in Digital Images Using Classification Tree”. The study reviews and summarizes image processing techniques for tomato plant diseases and classification tree algorithm is used for classification. The limitation of this research is no neural network based algorithm is used for classification and also the accuracies between the algorithm is also not considered [6].

3.1. SUMMARIZED FINDINGS

All proposed approaches were image-processing-based and is highly based on K-Means clustering technique and Artificial Neural Network (ANN). The approach is composed of four main phases; after the pre-processing phase, the images at hand are segmented using the Kmeans technique, then some texture features are extracted in which they are passed through a pre-trained neural network. Different researchers have worked on multiple classification algorithms namely ID3, Support Vector Machine (SVM), Naïve Bayesian from which SVM has been concluded to be the best among the all.

SVM was giving a precision of about 93% but, we wanted to make it more accurate and increase the speed of processing. That's why we tried rebuilding our model and made the Convolutional Neural Network (CNN) model from scratch using TensorFlow library Keras by defining convolutional layers which converts an image into 2D Convolve which further by using feature_extractor_layer gives 95% accuracy. This accuracy can also be increased by fined tuning but we focused on increasing the speed. Along with that above studies worked only on the disease prediction and there is no mention of the pesticides to be used to cure the disease.

4. METHODOLOGY

Due to the lack of some conditions like research scientists, agri-tech doctors availability, and high cost to get those services from them in villages, our team planned to achieve a high-tech system. This system can easily detect disease to plant by capturing images of the plant leaf and recommend Crop Name, Disease name, Confidence (Percentage of the Healthy Part of Leaf) and Pesticides to be sprayed those diseases affected crop just in one-click on Application. But to achieve such great accuracy, we had to implement some necessary image processing techniques and undergo artificial intelligence on our system to automatically detect disease to plants whose leaf image is given as input. To achieve the same, we have to undergo through the various process which is given as follows:

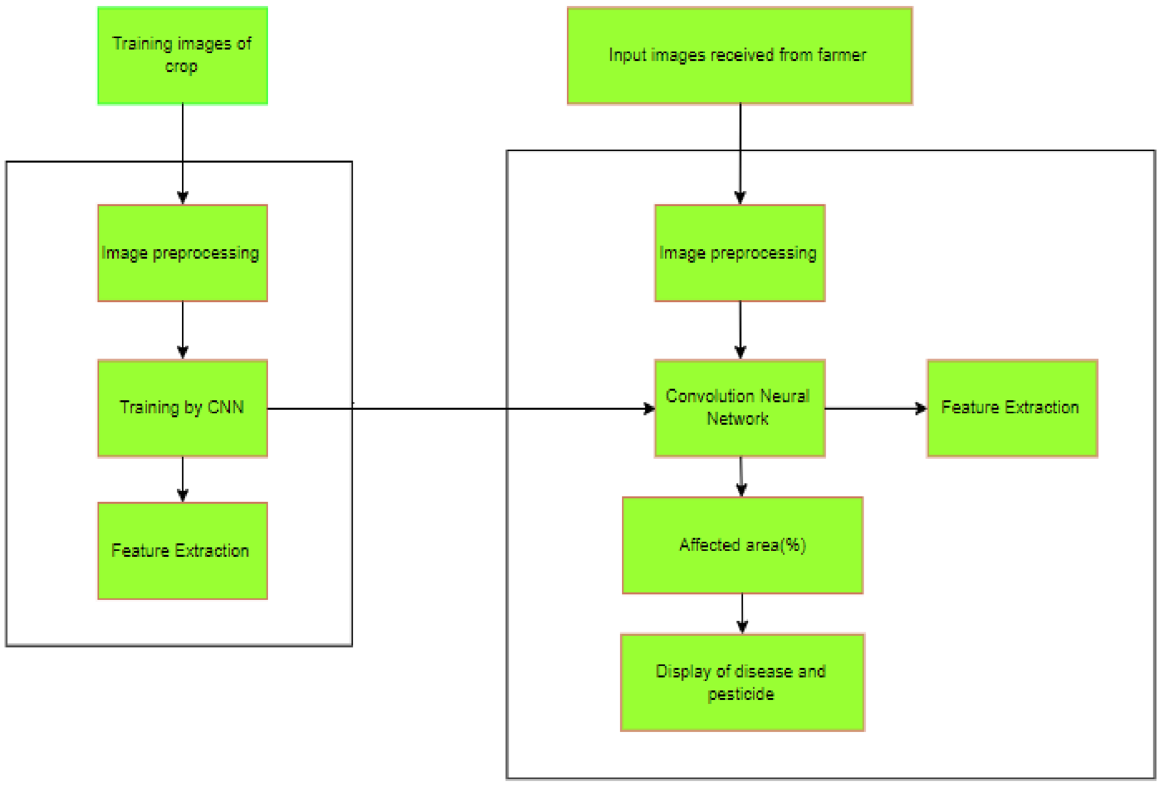


Figure1:ProcessingArchitecture

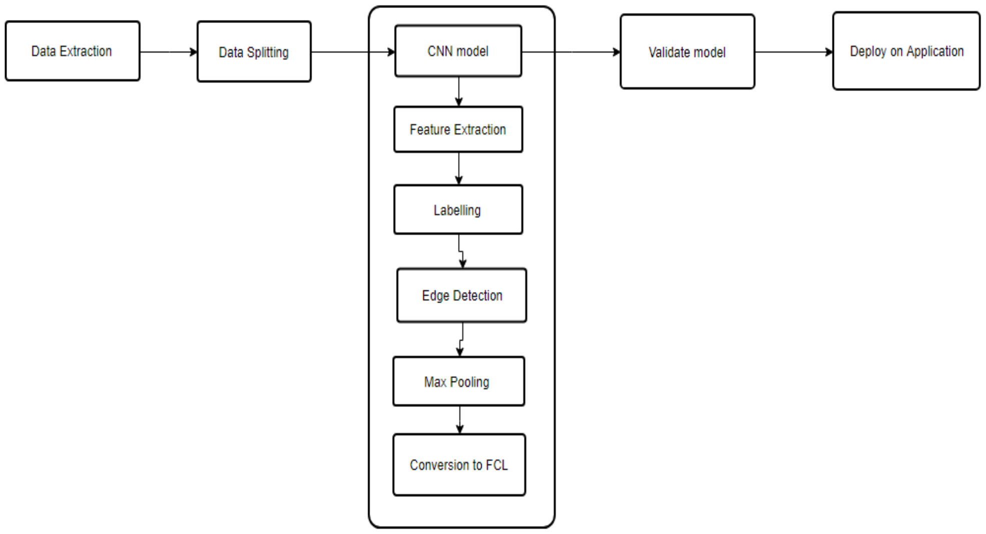


Figure 2: Process Flow Diagram

4.1. Data Extraction: There are various datasets publicly available on the Internet that can be used to implement our disease detection but when comparing all datasets to achieve better accuracy, we choose the most recommended PlantVillage dataset available on Kaggle and Google API engine but due to some problem it is removed from google API website. It is a public dataset of 54,305 images of diseased and healthy plant leaves by some officials. The images cover 14 species of crops which can be listed as follows: apple, blueberry, cherry, grape, orange, peach, pepper, potato, corn maize, raspberry, soybean, squash, strawberry and tomato. It contains images of 17 basic diseases, 4 bacterial diseases, 2 diseases caused by mold, 2 viral diseases and 1 disease caused by spider mites. 12 crop species also have healthy leaf images that are not affected by any disease. The Plant Village database is not optimum for training deep learning models like CNN because of similar backgrounds and lighting conditions and to get optimization to have to apply some necessary mechanisms. We used here multiple approaches to get rid of this problem like image segmentation to extract relevant features, applying random transforms to the extracted images to vary brightness, contrast, blur, and the internet-sourced images contained many different lighting conditions and backgrounds. For training deep learning models from scratch, it can be insufficient and previous studies have used data augmentation to multiply the number of samples multifold (Zhang et al., Barbedo, and Liu et al.) [7]. However, as we shall demonstrate, our model can achieve very high levels of accuracy with the dataset at hand.

4.2. Splitting the Data: Deep neural networks require two separate datasets to develop models. The first set, the training data set, is a collection of images that is used by the system to understand its hidden parameters, which including weights and biases, automatically. The validation set is also used to manually adjust hyper - parameters, and that are primarily settings that cannot be learned immediately during training. These include, for example, the learning rate, batch size, and network architecture. More information on hyperparameters can be found in Goodfellow et al. (2016) [9]. Because they are related to the problem, the dataset, and the model

architecture, the values of these hyperparameters are frequently set empirically. As a result, there are no good predefined values because they must be tuned based on the validation set's performance (in terms of accuracy). This means that information about the validation data is leaked into the model in an indirect way, resulting in an artificial ability to perform well on these 14 images (Chollet, 2017) [9]. As a result, the validation images should only be used to tune the hyperparameters; the test set, which is discussed in the following paragraph, is used to evaluate the model's performance. At the end of each epoch, the model being trained can be evaluated on the validation set, allowing the training process to be monitored and overfitting detected. The training and validation sets are derived from the same subdivided data source. The majority of the images for training were subjected to five different separation ratios, and both concluded that using 80 percent for training and 20 percent for validation was optimal for their data. We can also use cross-validation to divide the images into training and validation sets in a variety of ways. The dataset is divided into subsets that are used at random for training or validation. When the dataset is small, cross-validation is useful because it avoids any resulting bias caused by an arbitrary predetermined and fixed data separation. Once the hyperparameters are defined, additional training can be performed by combining the training and validation sets to benefit from a larger number of images.

5 IMPLEMENTATION

5.1. CNN Model/Architecture:

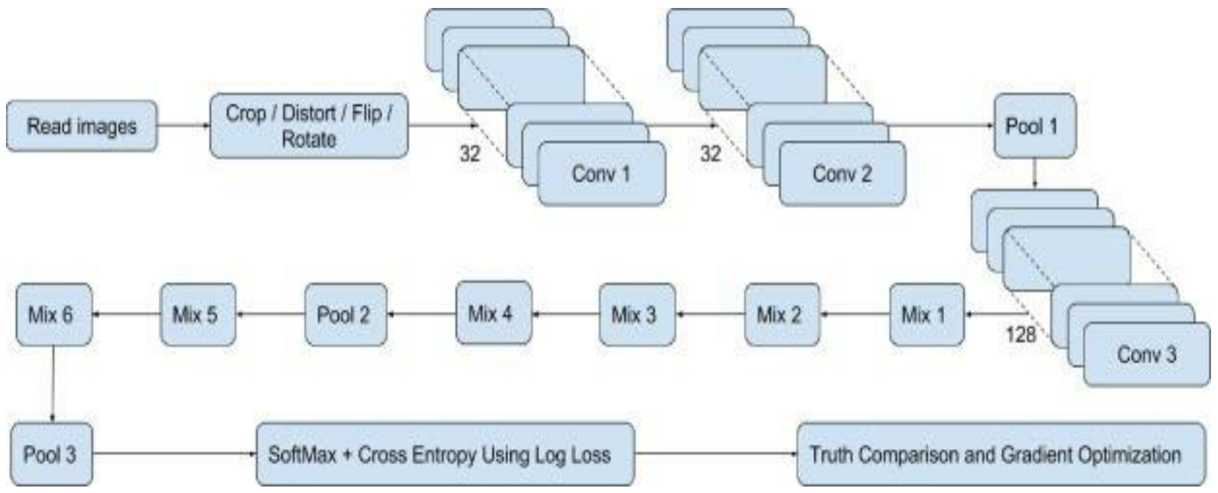

Read images
Crop / Distort / Flip / Rotate
32
Conv 1
32
Conv 2
Pool 1
128
Conv 3
Mix 1
Mix 2
Mix 3
Mix 4
Pool 2
Mix 5
Mix 6
Pool 3
SoftMax + Cross Entropy Using Log Loss
Truth Comparison and Gradient Optimization

Figure 3: CNN Model

The CNN model trained in this study is shown in above Figure. Once images are read, 255x255 pixel random parts of the images are extracted, and noise, distortion, flip or rotation transforms are applied. By controlling stride lengths (spacing interval for placement of the filters/masks), dimensions of masks, multiple convolution, and pooling steps are applied. Pooling involves the application of a mask to each pixel and then selecting a single value (eg. maximum) from with the mask. Mixing steps in the above Figure involves multiple convolution and pooling steps. By feeding the final output into a SoftMax function, a gradient optimizer is used to train the model weights. By introducing nonlinearities using a Rectified Linear Unit (ReLU) and random dropouts during learning, the model is able to automatically learn the decision boundaries required to classify images into one class or another. As the data proceeds through the network, the size of the filters and outputs changes (getting smaller progressing through the network), which allows for training for and detection of similar features with different scaling. The convolution filters applied over the whole images for training data mean that the CNNs are invariant to transformations of features such as rotation and translation. Furthermore, the pooling layers effectively make the CNNs tolerant to distortions in the features. The model is implemented using Python and TensorFlow, Training and validation runs were carried out on a hosted server at Google Cloud using Nvidia GPUs. In order to analyze the working of the model in detail, we pick out some plant leaves viz. Tomato, Potato, Corn and Orange, which include various diseases and healthy leaves, the symptoms are shown in below image. We have trained our model using F-CNN and used two data pre-processing techniques which are often necessary i.e. First, the images must typically be resized to match the size of the input layer of the CNN and hence we resized our image by 244x244 pixels which is also known as convolution on image using ConvNet and here we used DenseNet for multiscale features to implement on images. Secondly, Data to implement inside the Neural Network should be normalized, hence we preprocess our images by normalizing the pixel values to be in the `[0, 1]` range which was originally in '[0,255]' range.

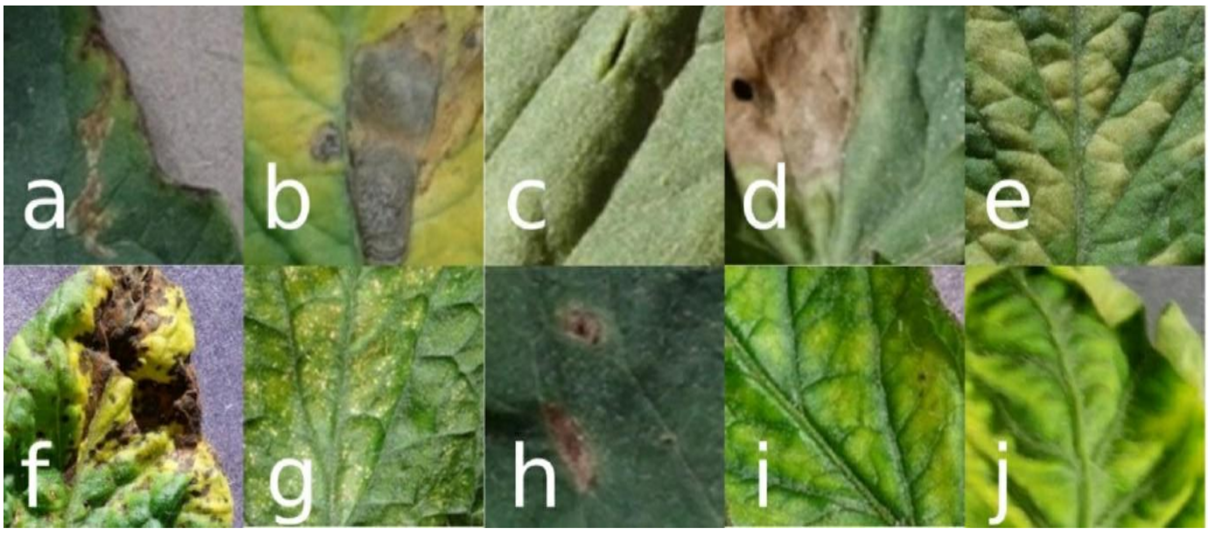


Figure 4: Leaf Features

5.1.1. Feature Extraction: Feature extraction is done using keeping a linear classifier on top of the feature_extractor_layer with the Tensorflow Hub module. We have not used fine tuning in our project for better accuracy rather focused on speed that's why we used a non-trainable feature_extractor_layer you can use fine-tuning as per your need. Feature Extraction is just getting a new image in a squeezed format by resizing according to the need of input to the CNN model. It is based on Filter we use for our project. Filter used in CNN model is automatically generated by keras using tensorflow and finally generates filtered images using the same.

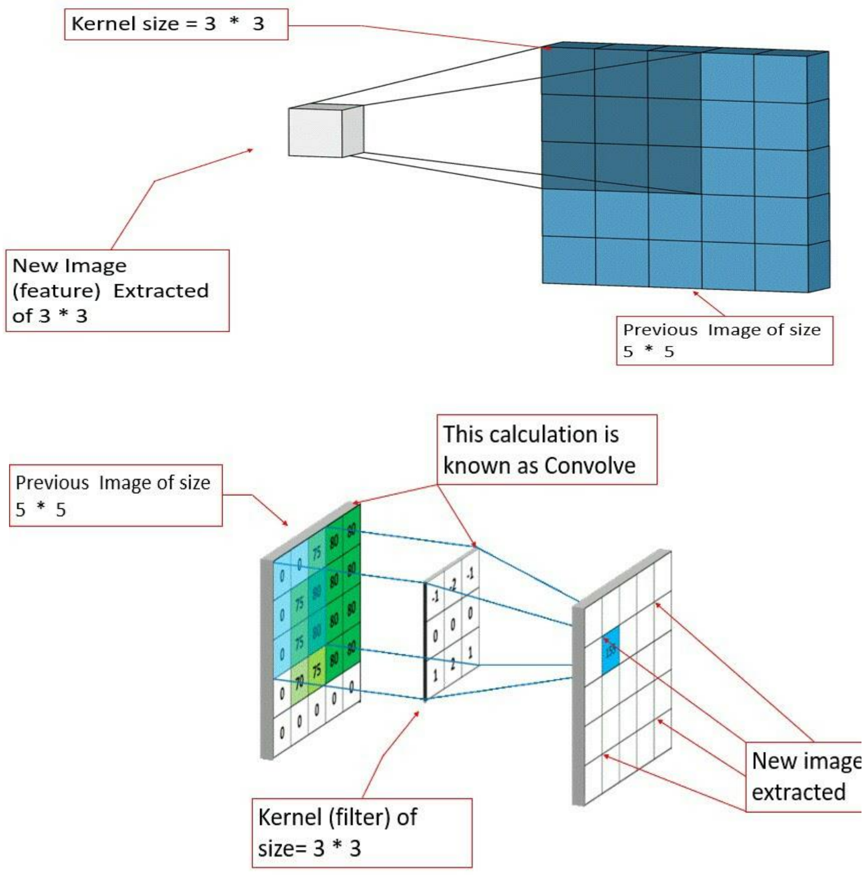

Kernel size = 3 * 3
New Image (feature) Extracted of 3 * 3
Previous Image of size 5 * 5
This calculation is known as Convolve
Previous Image of size 5 * 5
New image extracted
Kernel (filter) of size= 3 * 3

Figure 5: Feature Extraction

5.1.2. Labelling: We have loaded here a mapping from category label to category name describing the classification of plants distributions by their respective diseases. You can find this in the file categories.json file loaded in our model as a labelling file. It's a JSON object which you can only read in with the json module. This will give you a dictionary mapping i.e. key-value pair distribution of the integer encoded categories to the actual names of the plants and diseases. Cherry___Powdery_mildew, Tomato___Tomato_mosaic_virus as examples and other different types of classes together such 38 classes are labelled and given as input to the tensorflow model used for our system.

5.1.3. Feature Edge Detection: Images generally contain various features like some part of images are of no use and to get such useful part out of input image, Generally all classification models like CNN use Feature Edge Detection technique which separates the image by detecting the boundary of images and it happens by using color separation or using some libraries in machine learning models like scipy. To get edges generally we have to give input as Gray scaled images to get clear edges which can be mapped as a good feature.

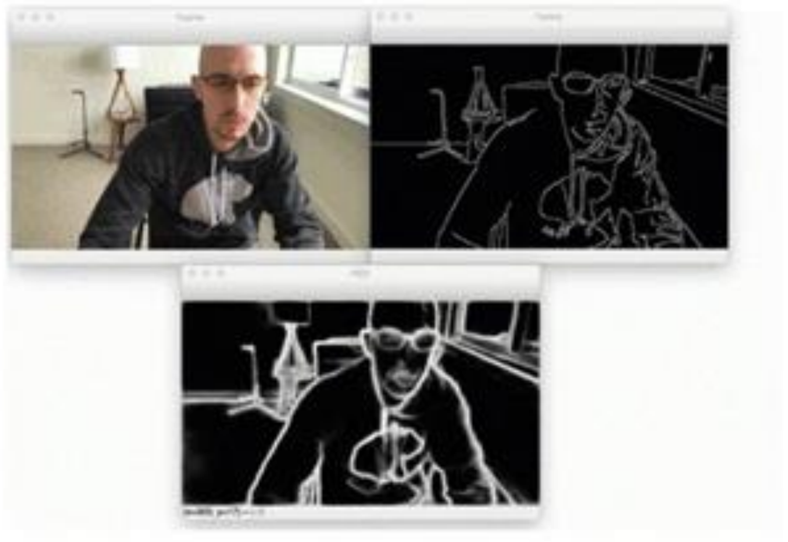


Figure 6: Edge Detection

5.1.4. Max pooling: Max Pooling is a technique that works in such a way that it produces the maximum image as output and is used in dense layers with activation functions to produce the maximum output, i.e. RLU (rectified Linear Unit), which produces y=max(x,0) as output. In our model, we used two layers: ReLU and Softmax. Softmax is the most accurate activation function, and it is used to generate K probabilities as output. The output layer employs the Softmax function.As a result, they are sent to flattening to be converted into 1D vector for further processing.

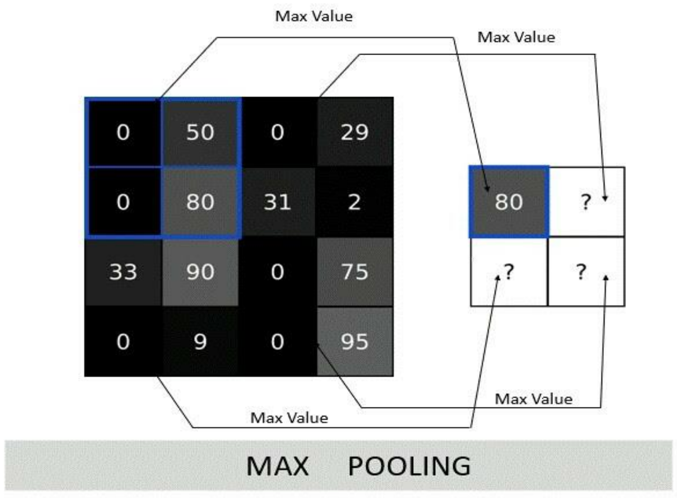


Figure 7: Max Pooling

5.1.5. Flattening and conversion to Fully Connected Layers (FCL): From previous steps in Deep Learning architecture, we gained our output as a 2D image trained with some methods; thus, flattening is the process of converting the image obtained, which is in a two - dimensional array, into a single-dimensional array, also known as Vector, which converts into Fully Connected Layers. These layers are interconnected and contribute to the best accuracy in the CNN model.

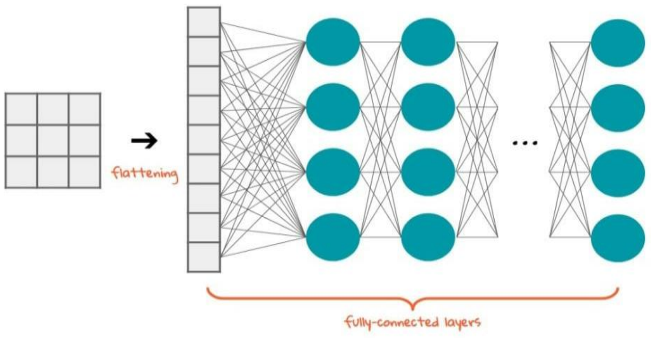


Figure 8: Fully Connected Layers

5.2. Train/Validate Model and Performance Metrics:

Let's take a glance at the training curves of the training and validation accuracy/loss when using the MobileNet V2 base model as a hard and fast feature extractor. The validation metrics are better than the training metrics, because layers liketf.keras.layers. BatchNormalization and tf.keras.layers.Dropout affect accuracy during training of the model using epochs. When calculating validation loss, they are disabled. To prevent overfitting, the Dropout layer randomly sets input units to 0. Inputs that are not set to 0 are scaled up by 1/(1 - rate) so that the sum of all inputs remains constant. Note that the Dropout layer only applies when training is about to True such no values are dropped during inference. When using model.fit, training is going to be appropriately set to True automatically, and in other contexts, you'll set the kwarg explicitly to True when calling the layer. It is also because training metrics report the average for an epoch in comparison with validation metrics areevaluated after the epochs used for training our CNN model, So validation metrics see a model that has trained slightly longer and also it is based on feature_extractor used while training [9].

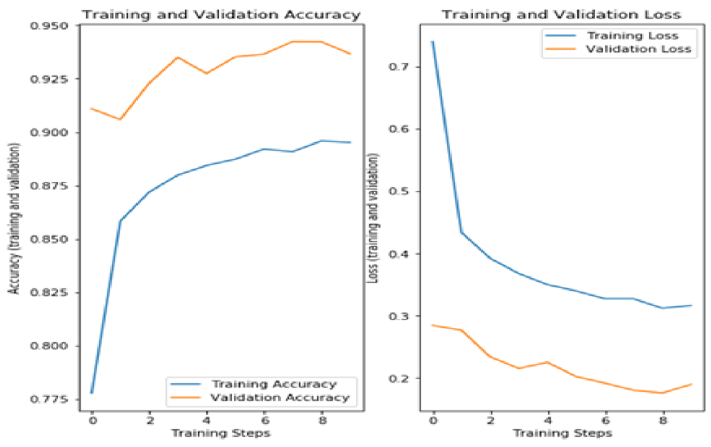


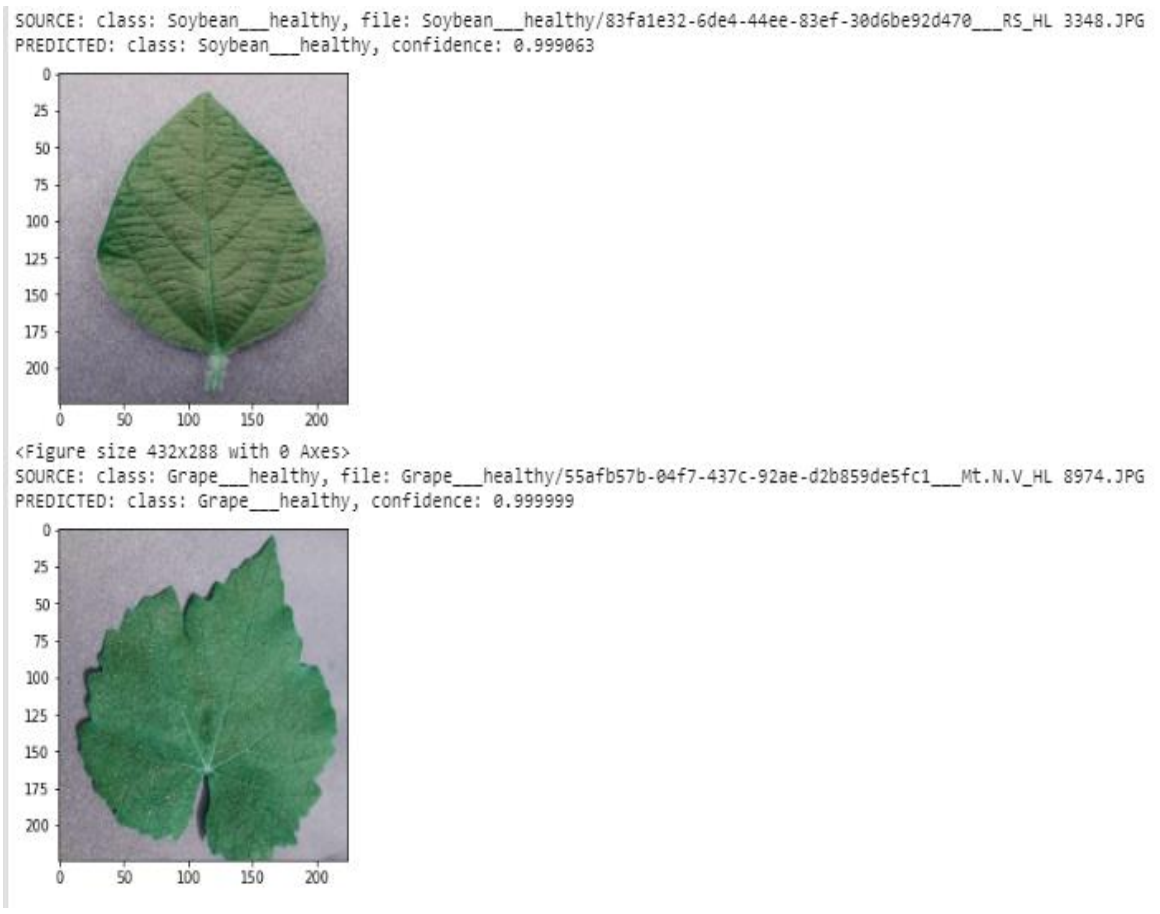
Figure 9: Performance Metrics

5.3. Loading/Balancing: We have got our model trained and accurate with training accuracy as 94% and testing accuracy as 87%. We can’t train our model again and again because it requires High GPU and takes around an hour to complete training of images so to use the model whenever we require loading the model in any form is necessary. Hence converted and stored model in the format for future implementation of project.

5.4. Deployment on Application: After getting a full-fledged model, We have easily converted it to a tflite form of model to load it into the connectivity between the Application. We are going to use Flutter based Application for better user experience of our system and for the same tflite is the based available format of model.It is done by extraction of concrete function of previous model and using keras concrete function converts model into standard Tflite model.

6. RESULTS AND DISCUSSION

6.1. Results for Trained Convolution Neural Network Model



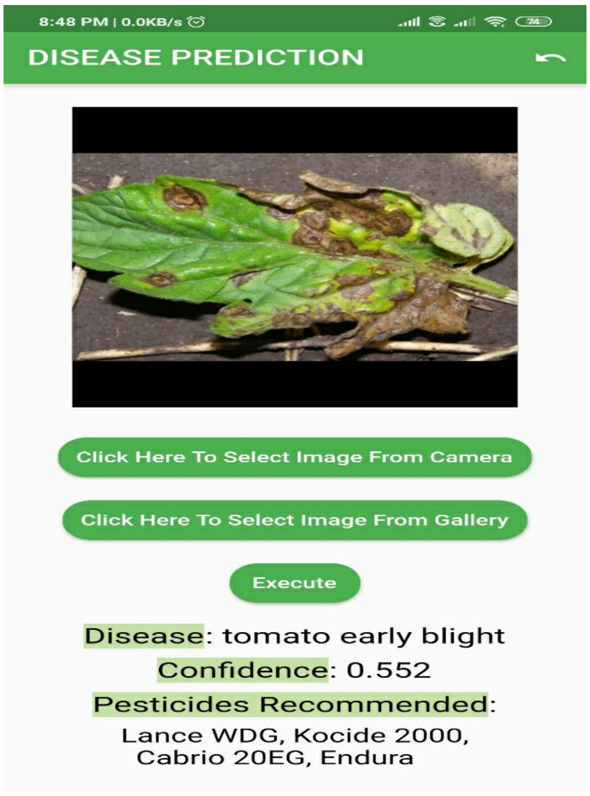
```
SOURCE: class: Soybean___healthy, file: Soybean___healthy/83fa1e32-6de4-44ee-83ef-30d6be92d470___RS_HL 3348.JPG
PREDICTED: class: Soybean___healthy, confidence: 0.999063
```

```
<Figure size 432x288 with 0 Axes>
SOURCE: class: Grape___healthy, file: Grape___healthy/55afb57b-04f7-437c-92ae-d2b859de5fc1___Mt.N.V_HL 8974.JPG
PREDICTED: class: Grape___healthy, confidence: 0.999999
```

Figure 10: Output of Soybean and Grape Leaf

The above image consists of the testing output of Soybean and Grape Leaf from the trained model. It displays the output of soybean leaf as healthy with the confidence of 0.99. Also, the grape leaf is healthy with the confidence of 0.99. It indicates that for the above leaf images the model shows high accuracy.

6.2. App Output Page

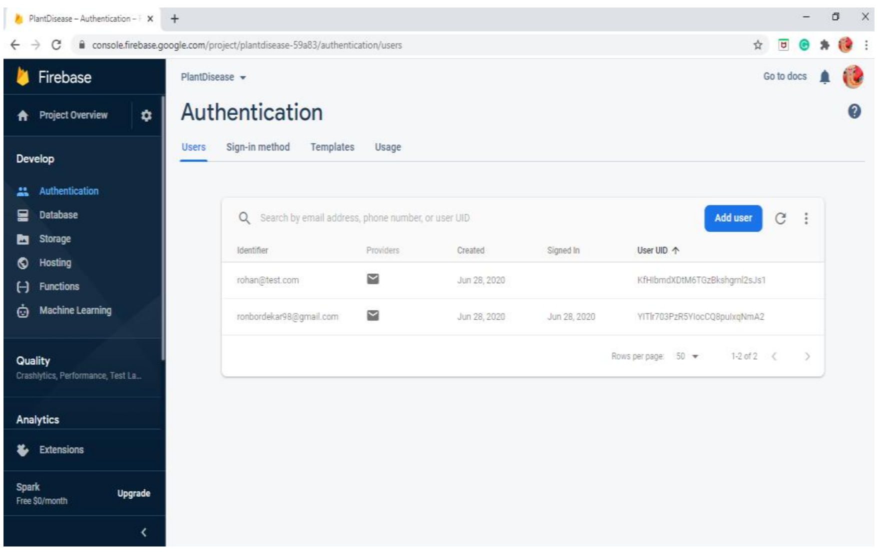

8:48 PM | 0.0KB/s
DISEASE PREDICTION
Click Here To Select Image From Camera
Click Here To Select Image From Gallery
Execute

Disease: tomato early blight
Confidence: 0.552
Pesticides Recommended:
Lance WDG, Kocide 2000,
Cabrio 20EG, Endura

Figure 11: App Output Page

The above image is the output page of the application. It consists of three buttons which are Select Image from Camera, Selects Image from Gallery, and Execute. On selecting the image from Camera/Gallery, the image gets displayed on the screen. On clicking the "Execute" button, the application processes the image and shows output on the screen as the disease of the leaf, its confidence, and pesticides recommended for curing the respective disease.

6.3. Database Console Output Page



Identifier	Providers	Created	Signed In	User UID ↑
rohan@test.com	✉	Jun 28, 2020		KfHIbmdXDtM6TGzBkshgml2sJs1
ronbordekar98@gmail.com	✉	Jun 28, 2020	Jun 28, 2020	YITlr703PzR5YlocCQ8pulxqNmA2

Rows per page: 50 1-2 of 2

Figure 12: Firebase Console Page

The above image is the Firebase Console Page which stores the credentials of the respective users and through which the user authentication takes place. Also, It stored the database collected by the userbase for better performance and service.

7. CONCLUSIONS AND FUTURE WORKS

An application for detecting the plant diseases and providing the necessary suggestions for the curing the disease has been implemented. The diseases specific to these plants were considered for testing of the model. By this method, the plant diseases can be identified at the initial stage itself and the pest control tools can be used to solve pest problems while minimizing risks to people and the environment. In order to improve disease identification rate at various stages, the training samples can be increased with the optimal features given as input condition for disease identification and pesticides management of the crops. In the future, Updates in the Application can be done like, The picture which is uploaded by the farmer in the app can be further used with the pictures in the training set and hence increasing the overall accuracy of the project. As we are recommending some pesticides in our research, in the upcoming works ML can also be applied to various pesticides, like training the pesticides and predicting them along with the amount required to cure the crop. As we are recommending some pesticides in our research, in the upcoming works ML can also be applied to various pesticides, like training the pesticides and predicting them along with the amount required to cure the crop. As this research consists of 14 crops for now, in future this application can be combined with all the plants as we get enough dataset as we can achieve the best accurate model by fine-tuning.

8. REFERENCES

[1] Dae Gwan Kim, Thomas F. Burks, Jianwei Qin, Duke M. Bulanon. (2008). Classification of grapefruit peel diseases using color texture feature analysis. (Vol. 2 No.3). Int J Agric & Biol Eng.

[2] Wang Jun and Wang Shitong. (2008). Image Thresholding Using Weighted Parzen-Window Estimation. (Volume 8 (5): 772-779). Journal of Applied Sciences.

[3] Prakash M. Mainkar1, Shreekant Ghorpade2, Mayur Adawadkar3. (2015). Plant Leaf Disease Detection and Classification Using Image Processing Techniques. (Volume 2, Issue 4). International Journal of Innovative and Emerging Research in Engineering.

[4] Garima Tripathi1, Jagruti Save2. (2015). An Image Processing and Neural Network based approach for Detection and Classification of plant leaf diseases. (Volume 6, Issue 4). International Journal of Computer Engineering & Technology (IJCET).

[5] Satish Kumar. (2016). Tomato plant disease classification in digital images using classification tree. International Conference on Communication and Signal Processing (ICCSP).

[6] Romana Tazeen, Shilpa H N, Usha P, Mrs. Jayanthi M G, Dr. Shashikumar D R. (2016). Image Processing System for Fertilization Management of Crops. International Conference on Computer and Information Technology (ICCIT).

[7] Parul Sharma, Yash Paul Singh Berwal, Wiqas Ghai. (18 November 2019). Performance analysis of deep learning CNN models for disease detection in plants using image segmentation. https://www.sciencedirect.com/science/article/pii/S2214317319301957

[8] Justin Boulent, Samuel Foucher, Jerome Theau, Pierre-Luc St-Charles. (23 July 2019). Convolutional Neural Networks for the Automatic Identification of Plant Diseases. https://www.frontiersin.org/articles/10.3389/fpls.2019.00941/full

[9] TensorFlow (https://www.tensorflow.org/tutorials/images/transfer_learning)

[10] Wikipedia (https://en.wikipedia.org/wiki)