International Journal of Psychosocial Rehabilitation, Vol. 24, Issue 03, 2020
ISSN: 1475-7192

Identification of Leaf Disease based on Artificial Intelligence

[1]G Manoj Kumar, [2]S Girirajan

Abstract:

Now a days, identification of diseases in plants has become very difficult in agriculture field. Having malady in plants are quite natural but if correct measures and care isn't taken then it causes adverse effects on plants. These effects may reduce the product quality as well as productivity. Since agriculture has been contributing a lot to the Indian economy, it is important to automate the identification and detection of plant diseases. In our paper, the ML model makes use of Transfer Learning. We used Teachable Machine, a web-based tool that makes creating ML models fast and easy. The new version Teachable Machine 2.0 makes AI easier for everyone. It builds a light weight network based on MobileNetV2.It performs Deep Learning tasks directly on the clients for better privacy and timely response (Browsers). Along with advancement in Computer Vision and increasing smartphone use made possible via Deep Learning, paved the way for smart-assisted disease diagnosis. The trained model achieves an accuracy of 99% on testing dataset.

Keywords: *Transfer Learning, TensorFlow, MobileNetV2, Teachable Machine V2.0, Deep Learning, KNN Classifier, Keras, POSTGRESQL, Flask*

I. Introduction

Agriculture being the backbone and important source factor of income for Indian economy, it contributes about 17% to the total GDP. So, there is a huge opportunity for farmers to grow wide varieties of crops on a major scale. Since plants are exposed to outer environment, they are highly prone to diseases. Mostly, the impact of plant disease is on Food security. The plant diseases reduce the quality as well as the production of food. So, there is a need for management to take care of protecting the crops in which these issues are rapidly growing. Generally, the plant disease can be controlled by disease diagnosis and implementing the correct measures for that disease. The existing method for disease detection involves constant monitoring, observation by specialists for long period of time in laboratories. It requires more manpower that costs them high once if farms are massive. Though in some villages, farmers don't have proper facilities to contact specialists and mobilize details to them. In such a case, our proposed technique helps farmers in automatically detecting the diseases in leaf just by simply showing the leaf to the live camera [1]. Then it suggests the right proportion of pesticides to be mixed and use for further spreading. This application makes the process simple and easy to use.

[1] Assistant Professor, SRM Institute of Science and Technology, Dept of CSE, Chennai, Tamil Nadu, India
[2] Teaching Associate, SRM Institute of Science and Technology, Dept of CSE, Chennai, Tamil Nadu, India

Received: 03 Dec 2019 | Revised: 21 Jan 2020 | Accepted: 18 Feb 2020

The leaf disease detection is just optical observation and constant monitoring by specialists which consumes a lot of time and cost ineffective. An automatic detection of diseases by simply capturing the plant leaves makes it easier and convenient to farmers [2]. Till now, several technologies have been emerged, however there's no application that can assist farmers to correctly identify the disease and provide remedies for the particular crop disease without having knowledge about its use.

In traditional leaf disease detection methods, CNN uses multilayer convolution to extract features and combine them automatically. It also uses fully connected layer, pooling layer and soft-max. Colour is an important feature in detecting the infected leaves to find the disease intensity. Traditional approaches considered grayscale, RGB images and used median filter for image enhancement. And also, segmentation for extraction of diseased portion. For detection of plant diseases, major techniques like Back Propagation Neural Network (BPNN), k-Nearest Neighbour (k-NN), Support Vector Machine (SVM) etc., are used [3].

II. Related Work

In general, Relevant data of images are found and lot of information on biological science. The identification of plant disease and growing a proper healthy plant has a huge impact and play an important role in agriculture. The farmers always use their naked eye to identify and detect a disease in the plant. The farmers can't continuously monitor on the large area of farms. They found this technique less useful and they don't know the non-native diseases. This system observes the various parts of the plant such as leaf, stem and root then note the change of characteristics and then informs the user regarding the disease. This paper provides the detection of the existing disease in the plants [13]. The Tomato crop is used for study in this paper. They have classified the tomato crop into 6 classes. Class one is based on healthy and unhealthy. The remaining five classes are on disease based on fungal, bacteria and virus and they also have the combined features such as colour, texture and shape features are obtained from the healthy and unhealthy plants. The results showed that classification has obtained a very good accuracy in the image classification and image recognition. There are many types of techniques for classification which are also used for image classification and image recognition. The classification techniques used are Adaptive neurofuzzy, Neural Networks, Genetic algorithm. The image classification is done on Support vector mechanism.so these are the techniques used for image classification and image recognition [14]. Disease on paddy crop is identified based on two types. Here they use the RGB image sample as input and then they compare it with the standard images. The standard images compare the 100 samples of healthy leafs and the two types of diseases in the training process. Later the images are verified using a histogram and the diseased part is then identified. The identified part is then separated. The histogram performs the distance calculation method to identify the leaf diseased area. In this project we aimed on the image processing technique. In the further project we can identify all kinds of diseases [15]. The tomato plant leaf disease detection and classification algorithm by Convolutional Neural Network (CNN) model and Learning Vector Quantization (LVQ) algorithm were used an input dataset gathering of around 500 tomato leaf images with the various four types of disease symptoms. We have also used an automation of CNN for the leaf feature identification and classification. The colour information is used through RGB component for leaf disease detection and classification. In this paper the filters are used through three channels by using RGB component. The output feature has been used through LVQ vector of convolution part for training the process. These experiment results show that the above process successfully

recognizes the four different times of tomato leaf disease identification [16]. The ML algorithm to represent large scale of dataset by identifying the disease present in a plant are proposed based on Random Forest algorithm which is a type of ML algorithm. It is used to find the anomalies which means separation after identification of disease between healthy and unhealthy leaf. This paper contains various phases including dataset creation, dataset gathering, training the images. The dataset creation includes a set of images. In dataset gathering the images are classified into healthy and unhealthy by using the random forest algorithm techniques. For extraction, the features of the leaf are extracted through the Histogram of an Oriented Gradient (HOG). Finally, by using the ML algorithm we classified the large set of data and then detected the disease in the leaf in a colossal scale [17]. The image data set is gathered, segmented, feature extraction and classification are done by using support vector machine. There are total of 90 grape leaf images for both the training and testing used. In which 70 are used for training and 20 are used for testing. The gathered data set is taken as an input in the RGB format and then converted it into a grayscale format. Then the segmentation of the diseased plant is done using histogram method and thresholding is used. The feature extraction is then done from the input based on color, texture, shape and edge. Then the classification of the diseased plant is done by using an support vector mechanism. The output of the system is obtained in the GUI format with 89.90% accuracy [18]. The image segmentation has been done by using the hue-based system which contains the thresholding of data, HSV and RGB conversion. The feature extraction is done by the input images by performing the texture analysis based on contrast, homogeneity, mean, variance. The segmentation of image is done by the support vector mechanism on the set of supervised data set. Then the output is generated by segmenting the training data for diseased plant after being the feature extraction [19]. The three different plants used such as Pomegranate plants, tomato plants and Okra plant. They have set of total 500 diseased Pomegranate images, 200 diseased tomato images and Set of total 79 diseased and non-diseased okra leaf images. The ML algorithm used in this project are K-Means algorithm. It is used in dataset segmentation for the identification of the diseased plant. K means clustering is done based on the color of the image as similar colors are grouped together. As for the classification, they used different types of techniques to classify they are Neural Network for pomegranate plant, support vector mechanism for tomato plant and naive Bayes classifier for okra plant. This project showed us the great accuracy for classification like 90% accuracy for pomegranate plant, 98% by using Laplacian function, 87% by naive Bayes for detecting and classifying images [20].

III. Proposed Leaf Disease Identifier

Recently several Javascript based deep learning frameworks have emerged, making it possible to perform deep learning tasks directly in browsers [1]. Teachable Machine is such a browser application that we can train the model with own webcam to identify and recognize the images, objects, audio etc., All training is done within the browser using the deeplearn.js library. It is a fast and easy way to create Machine Learning models for apps, websites etc., Today Google has number of pretrained models on Tensorflow official website. MobileNets is one of the pretrained models on Tensorflow. The teachable Machine relies on a pretrained image recognition network called MobileNetV2. The depthwise separable convolution [3] is replaced by expansion convolution, depthwise convolution and projection convolution (Inverted Residual structure), which is a light weight model used for image classification. The main idea of using this concept is that it can be extended to a greater number of crops for identification of leaf diseases where MobileNets [2] use less regularization and data

augmentation. And also, gives better accuracy which falls in the region of 90%-99%. One of the main advantages of this model is, it doesn't require more number of dataset for training the model.

3.1 MobileNet Architecture

3.1.1 MobileNetV1

The main idea of mobilenet V1 was its convolution layer which are very important for computer vision applications and quite expensive so thhey are replaced by depth wise Separable convolutions.

In the Mobilenet V1, there are two types of layer. They are:

i) The first layer is Depth wise convolutions.

ii) The second layer is 1x1 convolution layer which is called as point wise convolution.

Here RELU6 is also used for comparison. Normally RELU6 is used for its correctness.

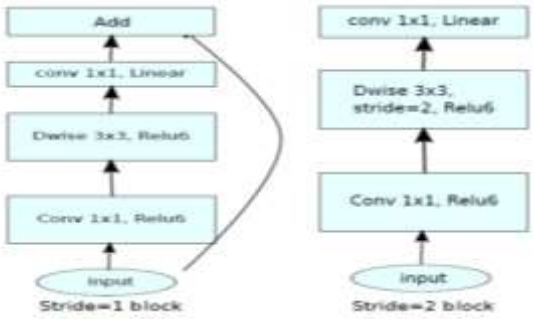


Fig 1 MobileNet V1 Architecture

The depthwise convolutions and the point wise convolution together is known as depthwise separable convolutions.

3.1.2 MobileNetV2

Today, Google is pleased to announce the availability of MobileNetV2 to power the next generation of mobile applications (Computer Vision) [6]. It is a significant improvement over MobileNetV1. It is released as a part of Tensorflow-Slim image classification library.

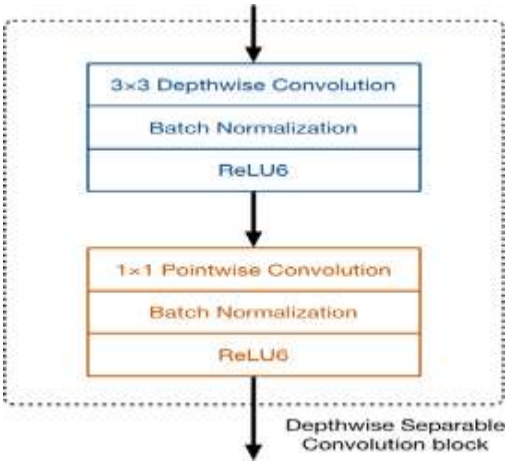


Fig 2 MobileNetV2

In MobileNetV2 there are 2 types of blocks.

i.) Residual block-with stride of 1.

ii.) Block with stride of 2 for downsizing.

There are 3 layers for both the above-mentioned blocks. The 1st layer is 1×1 convolution with ReLU6, 2nd layer is depthwise convolution and 3rd layer is 1×1 convolution but without non-linearity.

MobileNetV2 is built from some concepts involved in MobileNetV1. It is nothing but the extended version of V1. It introduces two new features to the architecture.

i.) Linear Bottlenecks between layers.

ii.) Shortcut connections b/w bottlenecks.

The basic structure is,

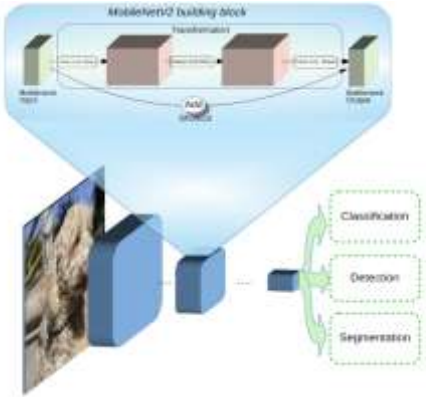


Fig 3 MobileNetV2 Architecture

Overall, the V2 models are faster for some accuracy across entire latency spectrum. These models require 30% fewer parameters, use 2 times less operations and about 30%-40% faster achieving higher accuracy.

3.1.3 Comparison between MobileNet V1 and V2

MobileNet V2 is an update version of V1 in terms of its performance which makes it most efficient and powerful.

Version	MACs(millions)	Parameters(millions)
MobileNet V1	569	4.24
MobileNet V2	300	3.47

Table 1: Performance of MobileNet V1 and MobileNet V2

The lower the Macs the better the V2.Macs also measure how many calculations are performed on a multiply-accumulate operations on a single 224×224 RGB image.

Even in terms of the accuracy the MobileNet V2 is better than V1:

Version	Top 1 accuracy	Top 5 accuracy
MobileNet V1	70.9	89.9
MobileNet V2	71.8	91.0

Table 2: Accuracy of MobileNet V1 and MobileNet V2

By Seeing the above result we can say that V2 is twice as fast as V1.On a mobile Device V2 works very well because the memory access is limited.

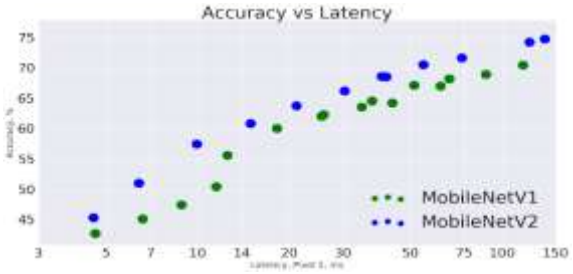


Fig 3 Comparison of MobileNetV1 vs MobileNetV2

3.2 Transfer Learning

The teachable machine uses the approach of transfer learning. It is basically storing the knowledge gained from previous tasks and can able to apply the knowledge on completely different tasks but related to predict the outcome. For example, if we know how to ride a motor cycle, we can able to learn how to drive a car. In most of the cases, we don't always learn the basics from scratch when try to learn some new topics. We make use of the knowledge what we have gained from past and apply here to learn something new [7]. All the traditional approaches are designed with conventional machine learning and deep learning algorithms. These models are trained to solve some specific tasks only. If there is any change in the feature space distribution, the model has to be built from scratch. So, to overcome these types of issues transfer learning is used. The learning of new tasks depends on the previous learned tasks during which the learning process can be way faster, accurate and the best part of this approach is it requires less amount of training data. So far, the deep learning models which gave good accurate results and performed very well requires a huge dataset containing millions of labelled images along with cluster of GPUs to train the models. But with google's advanced and updated Teachable machine v2.0, you can able to train a model with only few hundreds of images say 300 and a system with CPU under a minute. It is quite interesting that a machine can learn weights and classify objects within a short period of time[11].

3.2.1 Implementation of TM V2.0 using transfer learning model

There are two approaches that are generally used in case of neural network models. The first approach is, stripping out the last classifier layer and fitting our own classification layer. Here we train only this classification layer with our images. This does very well at classifying the images into hundreds of different

classes. For this situation, the base model acts like a fixed feature extractor by making a portrayal of a picture that has caught commonly important highlights. When we come across second approach, it goes one step further and permit the weights of the base model to be changed during the training time. This is generally referred as 'calibrating' or 'fine-tuning' the model.

Teachable Machine V2.0 uses the first approach that we discussed above. It treats the base model as fixed feature extractor. We can go further and know how a transfer learning model is implemented by Google TM V2.0.

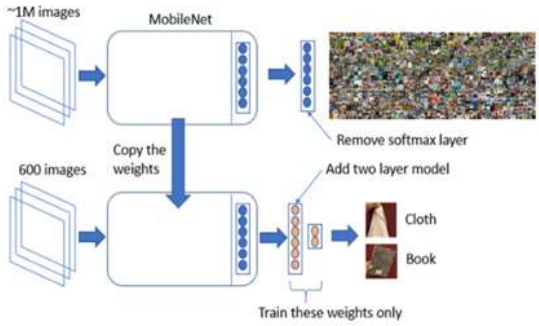


Fig 4 Implementation of TM V2.0 using transfer learning model

i) Take the MobileNet V2 model which is 28-layer conventional Network model to classify images.

ii) Strip off the softmax layer of this base model and set the output as one tesnor representation of images.

iii) Next, make a small model with just 2 layers.

a.) One dense layer.

b.) One softmax layer with same number of units that we require to classify the images (i.e. number of classes).

One of the important points to remember is that, the dense layer must accept the similar input as that of the output of mobileNet V2.

iv) Take images and convert them into its tensor using MobileNet. This step is more important beacuse we use tensors to train our model.

v) Train the 2-layer model alone with say 50 epochs, by making use of the prepared tensor training data.

vi) Make a joint model by appending MobileNet V2 and our 2-layer model.

vii) Use the model to predict new things. The entire model can be fit very well in the browser because mobilnet is so small (approx 2MB). The best part of this is, no image needs to leave the browser to train. All these models are written directly in javascript by the help of Tensorflow.js and gives the architecture some properties.

Received: 03 Dec 2019 | Revised: 21 Jan 2020 | Accepted: 18 Feb 2020

a.) Training is security safeguarding as images don't need the browsers train.

b.) Inference can be quick since it is serverless.

c.) There is no need to install any library.

d.) Since the model is coupled with transfer learning approach,new model can be trained with few minutes approach.

The methodology we discussed above clearly works incredible. If we ask ourselves that whether fine-tuning can help in any way to train inner neurons, the answer would be yes. The visual decathlon benchmark investigated in their paper [11] embarks to answer how extraordinary transfer learning techniques analyse in their performance. This paper evaluates the performance of different transfer learning approaches on various datasets like VGG, CIFAR100 etc., by making use of base model as Resnet28 which is trained on ImageNet dataset. The two different transfer learning approaches we discussed above are highlighted in the below figure. These numbers represent the top 1 accuracy in classifying an image.

ix) As the results say, the fine tuning seems to outperform a fixed feature extractor consistently. This is because, the model learns some ImageNet features like differentiating between several types of cars, breeds of animals etc, however training a ResNet28 is very computationally expensive and time consuming.

3.2.2 Teachable Machine

As we already discussed about Teachable Machine (TM) in the above paragraphs, we will now have a deep insight about this. TM lets anyone to build their machine learning model without the knowledge of coding. It makes use of pre trained model called MobileNet. It learns to predict the result using own classes that were never seen before by MobileNet. This is done by using activations produced by pre trained models, which represent high level semantic features of the image that the model has learned. The way pre training is done is way different what we actually think. It is so effective that there is no need to train another neural network. Instead nearest neighbours' approach is used. While feeding an image through MobileNet, the image that is having nearest and similar activation is found. The kk nearest neighbours' approach is used because it becomes noisy when comes into practise. So, by doing this, an image classifier is built with small amount of training data within very less time. This process is all done within the browser itself. So, this trained model can be made interactive that works with high class of accuracy. TM is used to classify images but also sounds and poses. But our project is limited to classify images.

IV. KNN Classifier

Our image classification model works on KNN (K-Nearest Neighbours) classifier. So, before going to describe what happens under the hood of KNN classifier, we will have a look on image classification in a nut shell.

We can assume a function f(x), that takes input as an image (say Potato leaf) and returns the output as a text label of it. It can be understood by the following picture.

Fig 5 A Blackbox function that takes image as an input and displays its text label

But this is quite absurd that how a function can able to take image as an input. Now, we can go through next steps and clear this question.

4.1 Image classification with KNN

KNN is a simple supervised machine learning algorithm that is used to solve both classification and regression problems. But here we will focus on classification. Let's go deep and dive into the mathematical regions that are happening and how we can classify those images.

Let us represent two sets of points on an XY plane with one set colored red and other colored green. These set of points represented here belong to two different classes or groups.

If one new point comes into the picture and we want to predict which group it belongs to. It becomes easy from human intuition perspective if that point is leaning towards either of the groups. What if there are 'k' number of groups and lies somewhere at the mid-point having n-dimensions. It is ideal that machines don't understand this kind of contextual information. So, we make machines understand by quantifying meanings and pass them into the program. One of the basic mathematical computations that comes into play is Euclidean distance. It has proven performance on real world datasets and that is the reason why it is used as a common distance metric. For some other datasets, distance metrics like Manhattan distance or cosine similarity etc., gives better performance. We use Euclidean distance to find the distance between two points. It is described as the following.

$$d(p_1, p_2) = \sqrt{\Sigma_d \left(p_1^d - p_2^d\right)^2}$$

This equation can be generalized and can reduce to any dimension. For 2-dimension, the equation becomes,

$$d(p_1, p_2) = \sqrt{\left(p_1^1 - p_2^1\right)^2 + \left(p_1^2 - p_2^2\right)^2}$$

This equation is similar to the equation we use in coordinate geometry to calculate distances.

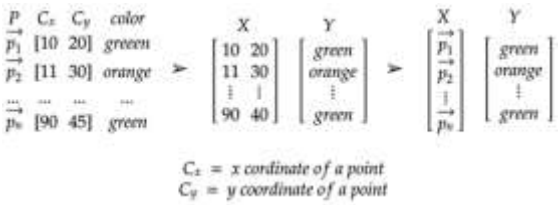
$$d(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \text{ where } p_1 = (x_1, y_1) \text{ and } p_2 = (x_2, y_2)$$

This KNN algorithm consists of four steps.

1. Calculate the distance between the new point and every other point in the plane.
2. Sort all the distances in ascending order.
3. Grab K least distances from the sorted distances (This is the reason behind calling it as K-NN).
4. Pick the one with the highest number of votes (classified groups) and assign the new point to that group.

4.2 Image classification intuition with KNN

Since there are 17 classes in our model, all points are represented in higher dimensional space. Each point in this space is represented as a vector containing a list of 17 numbers (each representing the coordinates in its plane). Since it is impossible to represent a higher dimensional plane here, we can have a look how this classifier works in 2D space.



Every point on a 2D plane is vector and the Euclidean distance can be derived from the following diagram.

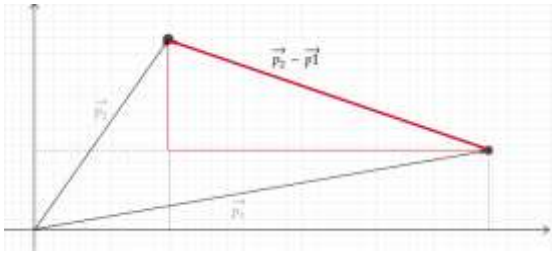


Fig 6 Geometrical representation of Euclidean distance equation

Let us assume a picture in our data set (say Corn) having a disease Cercospora of size 32*32. So, in total we have 1024 pixels. Inorder to represent this image in a way that machine can understand, we will have a vector of size 1024. These values will be the numbers of the color that pixel holds. The pixel's color can

be represented by three values(RGB channel), one is for red,another is for green and the other is for blue. So, for a RGB channel each has 1024 values and in total we have a vector of length 3072. The matrix would look like,

$$\begin{bmatrix} px_r^{1,1} & px_r^{1,2} & \dots & px_r^{32,32} & px_g^{1,1} & px_g^{1,2} & \dots & px_g^{32,32} & px_b^{1,1} & px_b^{1,2} & \dots & px_b^{32,32} \\ px_r^{1,1} & px_r^{1,2} & \dots & px_r^{32,32} & px_g^{1,1} & px_g^{1,2} & \dots & px_g^{32,32} & px_b^{1,1} & px_b^{1,2} & \dots & px_b^{32,32} \end{bmatrix}$$

We have discussed a question at the starting of this topic and the answer to that is, this is how we pass an image to the function. This is all done programmatically because machines cannot handle images and they should be represented in such a way the machine understands. So, they are represented in the form of large encoded vectors.

The image that we see in 2-dimension is nothing but a point we discussed before, but in higher dimensions. Though, the procedure of calculating the Euclidean distance is still valid for higher dimensions. The image here is represented as point and clustering is also valid for images. It is always hard to visualize in higher dimensions. So, it is recommended to assume in lower dimension (say 2D) and scale it up to a higher dimension. Then we can apply KNN classifier for these images. The training part can be explained in later parts of the section.

4.3 Video Classification

There is a quite difference between normal image classification and video classification. In our project the outcomes are predicted based on live video i.e. video classification. Here we train a Convolutional Neural Network (CNN) using keras. We know that videos are nothing but a series of images or individual frames. Most of the practitioners believe that the video classification is nothing but performing image classification for M number of times the individual frames are produced in a video. This approach produces some problems. We tried to improve our actual video classification results more accurate by implementing a quite interesting approach. This can be explained by the following way. We can see how an image classification performs,

i. Input an image to the neural network.

ii. Predict the outcomes from the neural network

iii. Choose the class/label with highest probability.

Let us also see what a naïve video classification can do,

i. Input each frame from a video to the neural network.

ii. Classify them independently to each other.

iii. Predict the labels and choose with highest probability.

If we try to implement this approach, we can observe a problem which is nothing but “prediction flickering”. There will a chance of flickering in the predictions. This obviously makes us confusing to choose the correct outcome among the predictions. To overcome this problem, we have a technique called “rolling prediction average”.

So, our video classification algorithm becomes,

i. Input each frame from a video to the neural network.

ii. Obtain the predictions of each frame from the neural network.

iii. Maintain a list of past N predictions.

iv. Compute the average of last N predictions and choose the class/label with highest corresponding probability.

v. Output the label for video classification.

The results of this approach can be discussed in the evaluation section.

Our model takes an advantage of TensorFlow 2.0 and keras deep learning libraries by importing "tensorflow.keras" package. We also used to scikit-learn, matplotlib and OpenCV. Scikit-learn is a library for ML, Matplotlib is for plotting and OpenCV for processing and loading the images. We set our initial learning rate to 0.001 and training epochs to 50 and default batch size. We scaled the pixel intensities to the range [0,1] and converted the data and class labels to NumPy array format. After, we create one-hot encode vectors for the class labels and split the data into the ratio training: validation to 7:2. i.e. 70 % for training and 20% for validation and remaining 10% for testing. The model automatically performs data augmentation by setting random image rotation to 15 degrees clockwise or anti-clockwise. Thereafter, we initialize our MobileNetv2 model and set it up for fine tuning. The MobileNet V2 model is instantiated with weights that are pretrained on ImageNet. Then we construct a fully-connected layer which contains SOFTMAX layers and connect it to MobileNet V2. Here the Convolutional weights of MobileNet model are made constant and only the fully connected layer will be trained with new weights i.e. the weights from our train dataset. Now we are ready to train and compile the deep learning model. We also generate confusion matrix to derive the accuracy. Then we plot training loss/accuracy history to generate loss, val_loss, accuracy and val_accuracy for inspection by outputting the plot to an plot.jpg image file. Then we save our model to disk. With the train.py script we train the 17 class (3 crops) leaf disease dataset by executing the commands.

V. Experimental Setup

This section focuses on the experimental setup for disease detection model using Teachable Machine on TensorFlow framework [4]. The implementation of this image classification model is separated into four stages.

- Dataset Gathering
- Training
- Exporting
- Deploying

5.1 Dataset Gathering:

Since our image classification model is based on supervised learning approach, we make use of labelled data to train the model. In our project the model is trained to predict 17 types of crop diseases present among three different crops i.e. Corn, Potato and Tomato. The labelled dataset of a particular crop required is made available free on the Kaggle website. Our trained model is limited to three crops for some good sight of accuracy. All the dataset downloaded is already labelled. The following is a glimpse of crop disease dataset.

5.2 Training:

MobileNets training is done in TensorFlow. The size of the input to this network is small. Individually, these crops have different types of diseases. So, we

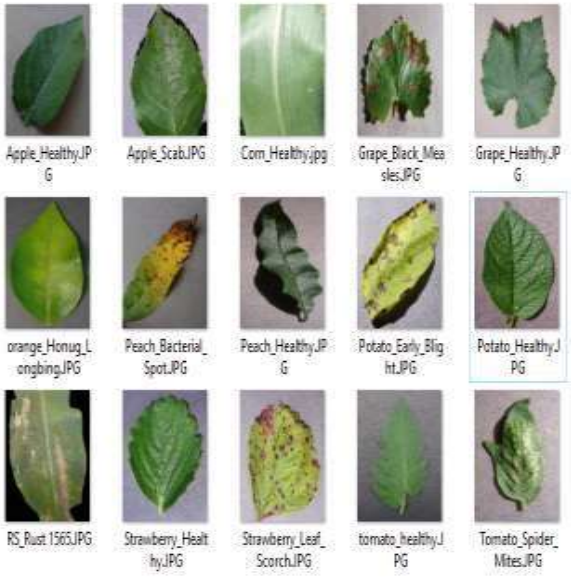


Fig 7 A glimpse of crop disease dataset

have labelled their class names by their disease names respectively. Since, there are a total of 17 diseases in these 3 types of crops (including healthy leaf), we have 17 types of classes. We can either upload the data from local device or through webcam. After uploading and naming the classes, we have to train the model. The training of the model is done in the browser itself. There are some advanced options like Epochs, Batch size and Learning rate etc., under the Train model option so that we can adjust accordingly. Choosing the correct parameters are very essential for the good accuracy of predicting the model. There is also an option called 'under the hood', to check what is actually going under this model training. The best part in training the model is uploading the training data through webcam. It records the images through some frames per second from the video. The process is continued for all the required classes and moved to the next stage.

5.3 Exporting

After training the model, we can export this model to use in the projects. It can be exported as a Tensorflow.js [7] model and can call it into any website or any application. It can also be converted to Tensorflow and Tensorflow.Lite [9]. The trained model is published to google servers except our training examples. When

we want to publish our model online, TM will generate a URL where we make use of this to deploy the model. Anyone who has the URL can able to use the model for their projects. But they cannot modify/change the model.

5.4 Deploying:

The trained machine learning model is a Tensorflow model which can be used with keras. So that the model can be called with URL where it is hosted online and make it respond accordingly. Our project is a simple web application that uses flask service and PostgresSQL database for extracting the remedies information for the disease present in the leaf. Since the remedies might change from day to day which includes some modern approaches of getting prevented from diseases. In order to update them, we used PostgresSQL database to store the information. Here we record a live video, that can be captured by webcam or uploading a recorded video. The processing takes place and it generate us the output as a video showing us the label of a disease written over the video as a text and displays us the remedy required to prevent the disease. The best part of this classification model is quite fast at recognizing the image frames, no matter what the size of a video it displays the result within the second it is shown.

5.5 Performance Evaluation

The following are the sample results of Potato crop relating to accuracy and precision.

From the below mentioned table we can conclude that if the leaf is healthy the model is pretty confident that it can predict correctly almost all the times (100% accuracy). If the leaf has a Early_Blight disease, it can predict correct only in 97% of the cases. If the leaf has Late_Blight disease, it can predict correct only in 95% of the cases.

Label	Precision	Recall	F1-score	Support
Potato (Early blight)	0.97	0.89	0.93	250
Potato (Late blight)	1.00	0.21	0.34	250
Potato (healthy)	0.95	0.83	0.91	250

Table 3: Performance Evaluation

VI. Results:

After testing the machine learning model, we conclude that the crop diseases are correctly classified with an accuracy value that lies between 90% and 97%. Though the resulting value might change since there is arbitrariness in the training process. On top of that if the crops are limited to two say Corn and Potato (7 classes) then we can expect higher precision. Convolutional Neural Networks are a emerging tool for processing and classifying images. Advances in the architecture of CNNs used and optimization of the training methodologies will help improve the accuracy of predicting the images. Making use of pretrained models helps us achieve more accurate results. So, it is always advised to use pre-trained models rather than building your own convolutional network architecture and training the model. The main drawback of this approach is it gives less accurate and false results unless you have a greater number of training data. Here in our case we don't require thousands of data to train the model. Only few hundreds (i.e less than 300) of data would suffice to train a classification model. Overall, we studied the existing approaches and methodologies used for detecting crop diseases and have come up with most efficient way to detect crop diseases. This helps farmers not to know the details of every crop disease and can easily detect the disease with the help of our classification model which can be extended to mobile application.

VII. Conclusion:

Data mining technologies has been introduced into agriculture field. Our project implements an innovative idea to detect and identify the affected crops and also provide remedies to be taken forward along with its volume proportional measure. By the use of Teachable Machine tool, the results are more accurate and faster. The inputs are to be shown in front of a webcam so that it records live and show us the outputs. This can be deployed and used as a Mobile Application where it is a good choice for agriculture community especially in villages and remote areas. As a future enhancement of the project is to introduce the IoT technology, where the temperature and humidity levels can be stored through sensors and this information is made useful for improving the accuracy. Introducing the text to speech conversion support of their personal choice for the better understanding will help farmers since they are the only one who has to understand what is actually happening around and everything in detail and to take decision to a step behind.

References:

[1] Edward Meeds, Remco Hendriks, Said Al Faraby, Magiel Bruntink, and Max Welling. 2015. MLitB: machine learning in the browser. PeerJ Computer Science 1 (2015), e11.

[2] Review: MobileNetV1—Depthwise Separable Convolution (Light Weight Model)

[3] Nitin R. Gavai, Yashashree A. Jakhade, Seema A. Tribhuvan, and Rashmi Bhattad "MobileNets for Flower Classification using TensorFlow", 2017 International Conference on Big Data, IoT and Data Science (BID) Vishwakarma Institute of Technology, Pune, Dec 20-22, 2017.

Received: 03 Dec 2019 | Revised: 21 Jan 2020 | Accepted: 18 Feb 2020

[4] Andrew G. Howard , Menglong Zhu, Bo Chen, Dmitry Kalenichenko, and Weijun Wang “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, arXiv:1704.04861v1 [cs.CV] 17 Apr 2017

[5] Shorav Suriyal, Christopher Druzgalski, and Kumar Gautam “Mobile Assisted Diabetic Retinopathy Detection using Deep Neural Network”, 2018 GLOBAL MEDICAL ENGINEERING PHYSICS EXCHANGES/PAN AMERICAN HEALTH CARE EXCHANGES (GMEPE / PAHCE).

[8] Viraj A. Gulhane, Maheshkuma R. H. Kolekar, "Diagnosis Of Diseases On Cotton Leaves Using Principal .

[9] Sharada P Mohanty, David P. Hughes and Marcel Salathe .”Using Deep Learning for Image –Based Plant Disease Detection”.

[10] Barbedo, Jayme Garcia Arnal, A new automatic method for disease symptom segmentation in digital photographs of plant leaves, European Journal of Plant Pathology, 2016, 1-16.

[11] G. Howard Andrew, Zhu Menglong, Chen Bo, Kalenichenko Dmitry, Wang Weijun, Weyand Tobias, Andreetto Marco, Adam Hartwig, "Mobilenets: Efficient convolutional neural networks for mobile vision applications", *CoRR*, 2017.

[12] "Xception: Deep learning with depthwise separable convolutions", *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[13] Huu Quan Cap, Satoshi Kagiwada,Hiroyuki Uga,Hitoshi Iyatomi,A Deep Learning Approach for on-site Plant Leaf Detection ,2018 IEEE 14th International Colloquium on Signal Processing & its Applications (CSPA 2018)

[14] Santhosh Kumar , B.K.Raghavendra -Diseases Detection of Various Plant Leaf Using Image Processing Techniques: A Review ,2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)

[15] L. Sherly Puspha Annabel, Member, IEEE, T. Annapoorani and P. Deepalakshmi - Machine Learning for Plant Leaf Disease Detection and Classification – A Review ,International Conference on Communication and Signal Processing, April 4-6, 2019, India

[16] Arya M S, Anjali K, Mrs.Divya Unni, “ Detection of Unhealthy Plant Leaves using Image Processing and Genetic Algorith with Arfunio”, 978-1-8386-4208-5/18, 2018 IEEE.

[17] Jobin Francis, Anto Sahaya Dhas D,Anoop B K, “ IDENTIFICATION OF LEAF DISEASES IN PEPPER PLANTS USING SOFT COMPUTING TECHNIQUES”, IEEE International Conference on Emerging Devices and Smart Systems (ICEDSS), 2016, page 168-173.

[18] Jagadish Kashinath Kamble, “ PLANT DISEASE DETECTOR”, International Conference On Advances in Communication and Computing Technology (ICACCT),978-1-5386-0926-2/18, 2018 IEEE

Received: 03 Dec 2019 | Revised: 21 Jan 2020 | Accepted: 18 Feb 2020

[19] Chit Su Hlaing, Sai Maung Maung Zaw, " Plant Diseases Recognition for Smart Farming Using Modelbased Statistical Features", IEEE 6th Global Conference on Consumer Electronics (GCCE 2017), 978-1-5090- 4045-2/17, 2017 IEEE

[20] G. Kambale and N. Bilgi, "A Survey Paper on Crop Disease Identification and Classification Using Pattern Recognition and Digital Image Processing Techniques," no. Acbcda, pp. 14–17, 2017.

[21] S. Raut and A. Fulsunge, "Plant Disease Detection in Image Processing," pp. 10373–10381, 2017

[22] K. S. Neethu and P. Vijay, "Leaf Disease Detection and Selection of Fertilizers using Artificial Neural Network," pp. 1852–1858, 2017.