Plant Leaves Disease Detection

Ayushi Pandey

Assistant Professor, Department of Computer Science Engineering, Inderprastha Engineering College Ghaziabad, India

---***---

Abstract - *The Agricultural sector plays an important role in sustainable economic growth and food security. But crop diseases often causes a great threat to achieve this growth. A successful outcome depends entirely on early detection and classification of plant diseases. This created various opportunities of new possibilities for research in this area. Nowadays, a lot of work is being done to recognize and classify plant diseases more precisely using neural networks. This paper presents a Convolutional Neural Network (CNN) model method for leaf disease detection and classification. I have modeled a CNN for automatic feature extraction and classification for classify the healthy and diseased leaves. The experimental results validate that the proposed method effectively achieves good accuracy*.

Key Words: **agriculture, Convolutional Neural Network, classification, feature extraction.**

1.INTRODUCTION

Plant diseases affect the growth and crop yield of the plants and make social, ecological and economic impacts on agriculture. Recent studies shows that about 50% crops get damaged due to diseases. Plant leaf diseases causes huge number of economic losses to farmers. Early detection of the diseases deserves special attention. The predictions are done according to the visible surface of plants leaves. Early Detection of diseases as soon as they appear helps for effective disease management. Traditionally detection is carried out by human experts or by means of mechanical cultivation. Detecting and Classifying Diseases in a timely manner is of the great importance so save the crops and plants leaves from further damaged. [1]. In recent years, CNN models have been widely used in image classification problems .Mercelin at al. [2] the Convolutional Neural Network model is created and developed to perform plant disease detection and classification using apple and tomato leaf images of healthy and diseased plants.The four-layer convolutional layers each followed by pooling layers model is designed. Two fully connected dense layers and sigmoid function is used for detection of presence of disease or not. Apple and Tomato leaf image dataset containing 3663 images used for training purpose which achieves an accuracy of 87%. The overfitting problem is identified and removed by setting up the dropout value to 0.2. Lee at al. [3] introduce a hybrid model to extract contextual information of leaf features using CNN and Deconvolutional Networks (DN). Konstantinos at al. [4] performed several pre-trained CNN models on a large open leaves dataset. Their studies show that CNN is highly suitable for automatic plant disease identification. Durmus at al. [5] used the Alex Net and the Squeeze pre-trained CNN models on tomato leaves from an open dataset to detect diseases and achieves good results.

1.1 *Convolutional neural network*

Deep learning is a class of the machine learning algorithms that consist of sequential layers. Each layer output used as input for the preceding layer. The three types of learning process are there unsupervised, supervised or semi-supervised. Representation learning algorithms makes optimizations to find the most efficient way to represent the data [5]. In Deep learning there is no need for division for the feature extraction and the classification because the model automatically extracts the features while training the model. It is used mostly in research areas such as image processing, image restoration, speech recognition, natural language processing and bioinformatics. CNN model is preferred as a deep learning method in this work. CNN, can easily identify and classify objects with minimal pre-processing, and successfully analyze the visual images and can easily separate the required features with its multi-layered structure. There is main four layers: convolutional layer, pooling layer, activation function layer and fully connected layer.

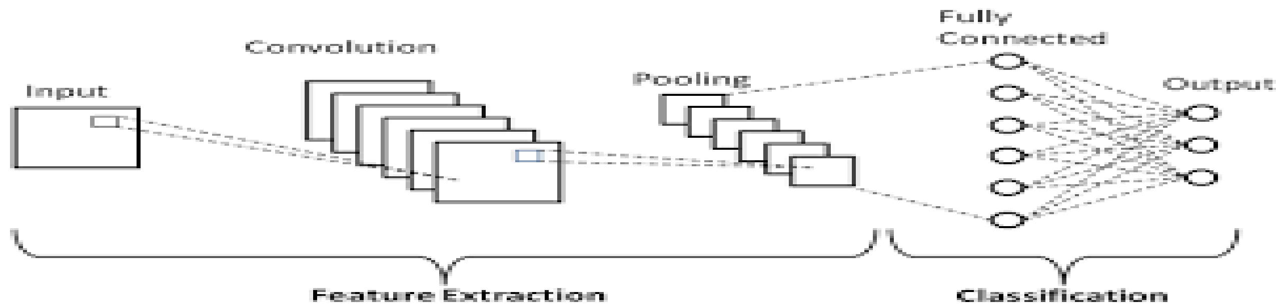

Fig1 CNN model diagram

A.Convolutional Layer CNN takes its name from the Convolution layer. In this layer, a series of mathematical calculations are performed to extract the feature map of the input image and input image is reduced to a smaller size using a filter. The filter is shifted step by step starting from the upper left corner of the image by defining the stride size. At each step, the values in the image are multiplied by the values of the filter and the result is summed up to form a new matrix with a smaller size is created from the input image.

B Pooling Layer The pooling layer is applied after the convolution layer. The size of the output matrix obtained after the convolution layer operations are reduced in this layer. Filters of different sizes can be used in the pooling layer, mostly 2x2 size filter is used. Several functions such as max pooling, average pooling can be used in this layer. In this work, max pooling filter with stride 2 has been applied. In Max pooling the largest value in the sub windows is selected and this value is transferred into a new matrix

C.Activation Layer In artificial neural networks, the activation function provides a curvilinear relationship between the input and output layers. It affects the network performance. Non-linear learning of the network can be achieved with the help of activation function. Different activation functions, such as linear, sigmoid, hyperbolic tangent, exist, but the nonlinear REL (Rectified Linear Unit) activation function is commonly used in CNN. In REL, values less than zero are changed to zero, while values greater than zero are unchanged by (1). $f(x) = 0$, if $x < 0$. x, otherwise. (1). Dually Connected Layer The matrix obtained in last, after performing all the operations convolution, pooling and activation, is fed into the fully connected layer as input. Recognition and classification are performed in this layer.

2. Proposed Methodology

In this proposed method we define a CNN model that helps to extract the features automatically and classify the plant leaves as healthy or infected. Steps to be performed during the research.

1.Image Acquistion:

The plant village dataset is taken for this research. Plant village dataset is available publicly. In this dataset is divided into 15 diseased classes and it contains healthy class of various plants leaves.

2. Label the data

In this part we assign label to the healthy and diseased images with the help of inbuilt function label binarizer in python. Healthy leaves assigned 0 label and diseased leaves assigned 1 label.

3. Define the CNN model architecture

In this part 5-layer CNN model is defined for 256*256 input images. CNN model consists of various filters at each input layer to extract the maximum features from the input images and at last classification is done at the end to identify the whether the leaves is healthy or infected.

4.Testing of model.

```
model.add(Conv2D(32, (3, 3), padding="same",input_shape=inputShape))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1024))
model.add(Activation("relu"))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(n_classes))
model.add(Activation("softmax"))
```

Fig2: CNN architecture

3. Experimental Results

1.Dataset collection: in this part we load our plant village dataset to machine. fig [3] shows the loading part of data on machine model.

```
[INFO] Loading images ...
[INFO] Processing Tomato_Late_blight ...
[INFO] Processing Tomato_Early_blight ...
[INFO] Processing Tomato__Target_Spot ...
[INFO] Processing Pepper__bell___healthy ...
[INFO] Processing Potato___Late_blight ...
[INFO] Processing Tomato_Bacterial_spot ...
[INFO] Processing Pepper__bell___Bacterial_spot ...
[INFO] Processing Tomato__Tomato_YellowLeaf__Curl_Virus ...
[INFO] Processing Tomato_Leaf_Mold ...
[INFO] Processing Tomato_Spider_mites_Two_spotted_spider_mite ...
[INFO] Processing Tomato_healthy ...
[INFO] Processing Potato___healthy ...
[INFO] Processing Potato___Early_blight ...
[INFO] Processing Tomato__Tomato_mosaic_virus ...
[INFO] Processing Tomato_Septoria_leaf_spot ...
[INFO] Image loading completed
```

Fig3: Dataset loading

2.Label assigning: in this part we assign the labels to our classes '0' assigned to unhealthy class and '1' assigned to healthy class. Fig4 shows the result which get after assigning labels to the class. labels are assigned with the help of label binarizer class

```
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
```

Fig 4: assigning labels

3.CNN model result

In this part I get output parameters for CNN model defined. The fig 5 shows the output of CNN model which defines various parameters values at the run time

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 256, 256, 32)      896
_________________________________________________________________
activation_1 (Activation)    (None, 256, 256, 32)      0
_________________________________________________________________
batch_normalization_1 (Batch (None, 256, 256, 32)      128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 85, 85, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 85, 85, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 85, 85, 64)        18496
_________________________________________________________________
activation_2 (Activation)    (None, 85, 85, 64)        0
_________________________________________________________________
batch_normalization_2 (Batch (None, 85, 85, 64)        256
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 85, 85, 64)        36928
_________________________________________________________________
activation_3 (Activation)    (None, 85, 85, 64)        0
_________________________________________________________________
batch_normalization_3 (Batch (None, 85, 85, 64)        256
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 42, 42, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 42, 42, 64)        0
```

```
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 42, 42, 128)       73856
_________________________________________________________________
activation_4 (Activation)    (None, 42, 42, 128)       0
_________________________________________________________________
batch_normalization_4 (Batch (None, 42, 42, 128)       512
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 42, 42, 128)       147584
_________________________________________________________________
activation_5 (Activation)    (None, 42, 42, 128)       0
_________________________________________________________________
batch_normalization_5 (Batch (None, 42, 42, 128)       512
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 21, 21, 128)       0
_________________________________________________________________
dropout_3 (Dropout)          (None, 21, 21, 128)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 56448)             0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              57803776
_________________________________________________________________
activation_6 (Activation)    (None, 1024)              0
_________________________________________________________________
batch_normalization_6 (Batch (None, 1024)              4096
_________________________________________________________________
dropout_4 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 15)                15375
_________________________________________________________________
```

```
_________________________________________________________________
dropout_4 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 15)                15375
_________________________________________________________________
activation_7 (Activation)    (None, 15)                0
=================================================================
Total params: 58,102,671
Trainable params: 58,099,791
Non-trainable params: 2,880
_________________________________________________________________
```

Fig5 CNN model overview with total parameters

4. Epoch counting: epoch counting provides us the details of losses occurred each time model trained. fig6

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 06 | June 2021 www.irjet.net p-ISSN: 2395-0072

```
Epoch 1/5
73/73 [==============================] - 78s 1s/step - loss: 0.2045 - acc: 0.9360 - val_loss: 0.8245 - val_acc: 0.8870
Epoch 2/5
73/73 [==============================] - 50s 681ms/step - loss: 0.1618 - acc: 0.9450 - val_loss: 0.5299 - val_acc: 0.8
977
Epoch 3/5
73/73 [==============================] - 60s 817ms/step - loss: 0.1445 - acc: 0.9516 - val_loss: 0.6714 - val_acc: 0.9
023
Epoch 4/5
73/73 [==============================] - 60s 818ms/step - loss: 0.1381 - acc: 0.9522 - val_loss: 0.9523 - val_acc: 0.9
043
Epoch 5/5
73/73 [==============================] - 61s 831ms/step - loss: 0.1124 - acc: 0.9609 - val_loss: 1.4147 - val_acc: 0.8
864
```

Fig 6 epoch counting

5. Accuracy graphs: there are two graphs which we got in our experimental result

1).Training and validation accuracy: this graph represents the accuracy at the trainning time at each epoch.

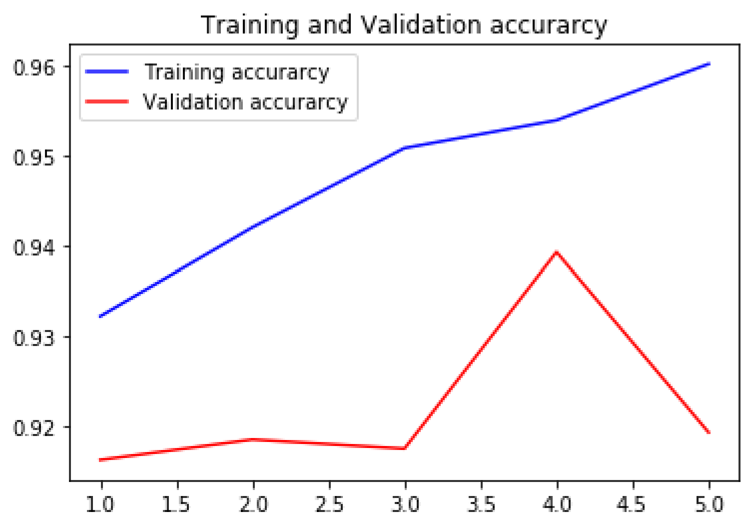

Fig 7Training and validation accuracy graph

2). training and validation loss graph: this graph represents the training and validation loss occur at each epoch.

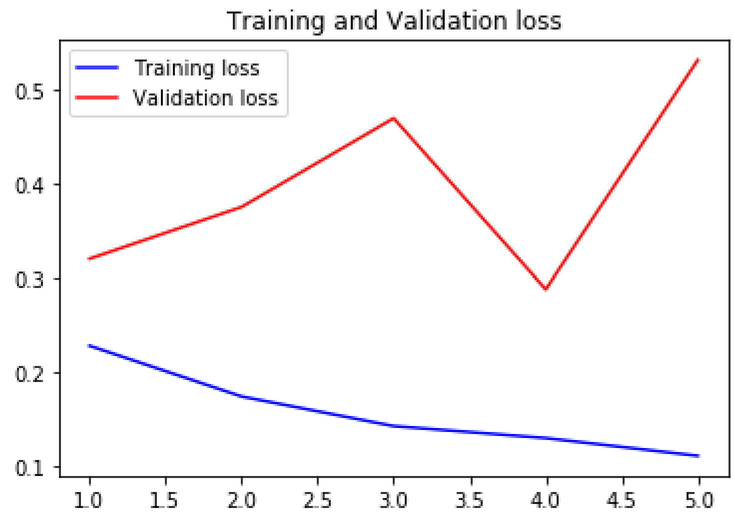

Fig 8 training and validation loss graph

6. Testing accuracy : this figure represents the testing accuracy and model achieves the 93.64% accuracy effectively.

```
[INFO] Calculating model accuracy
591/591 [==============================] - 2s 3ms/step
Test Accuracy: 93.64918351375108
```

Fig 9 test accuracy

4.Conclusion

In this paper, plant village dataset is used for diseases detection and classification. Approach based on Convolutional Neural Network. The dataset consist of more than 3000 images of various plants leaves images. Each input image matrix has been convoluted. RELU activation function and max pooling have been implied to the output matrix. The experiments have been carried out on healthy and diseased leaf images to perform classification. It is concluded that the proposed method effectively recognizes the leaf diseases. To improve recognition rate in classification process different filters or different size of convolutions can also be used by increasing the number of epochs and using SVM classifier for classification.

References

[1] H. Park, J. S. Eun and S. H. Kim, “Image-based disease diagnosing and predicting of the crops through the deep learning mechanism”, In Information and Communication Technology Convergence (ICTC), IEEE 2017 International Conference on, pp. 129-131, 2017

[2] Mercelin Francis, C. Deisy,” Disease Detection and Classification in Agricultural Plants Using Convolutional Neural Networks – A Visual Understanding”, 2019 6th International Confrence on Signal processing and Integrated network

[3] S. H. Lee, C. S. Chan, S. J. Mayo and P. Remagnino, “How deep learning extracts and learns leaf features for plant classification”, Pattern Recognition, vol. 71, pp. 1-13, 2017.

[4] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis", Computers and Electronics in Agriculture, vol. 145, pp. 311- 318, 2018.

[5] H. Durmus¸, E. O. Gunes ¨ ¸ and M. Kırcı, "Disease detection on the leaves of the tomato plants by using deep learning", In Agro-Geoinformatics, IEEE 6th International Conference on, pp. 1-5, 2017.

[6]Jagadish Kashinath Kamble," PLANT DISEASE DETECTOR", 2018 International Conference On Advances in Communication and Computing Technology .

[7] Chaowalit Khitthuk, Arthit Srikaew, Kitti Attakitmongcol , Prayoth Kumsawat," Plant Leaf Disease Diagnosis from Color Imagery Using Co-Occurrence Matrix and Artificial Intelligence System", 2018 IEECON Krabi, Thailand.

[8] Jingwei Liao, Mantao Wang, Zhouyu Tan, Weijun Gao, Yuchen Wang, Jie Zhang, Lixin Luo," The Design and Implementation of Plant Disease Spot Segmentation Algorithm Based on Improved CV Model", 2019 2nd International Conference on Safety Produce Informatization

[9] Pushkara Sharma, Pankaj Hans, Subhash Chand Gupta," CLASSIFICATION OF PLANT LEAF DISEASES USING MACHINE LEARNING AND IMAGE PREPROCESSING TECHNIQUES", 2020 IEEE

[10] Saradhambal.G, Dhivya.R , Latha.S , R.Rajesh," PLANT DISEASE DETECTION AND ITS SOLUTION USING IMAGE CLASSIFICATION", International Journal of Pure and Applied Mathematics Volume 119 No. 14 2018

[11]figure1imagewww.http\researchgate.net\Schematic-diagram-of-a-basic-convolutional-neural-network-CNN-architecture-26.ppm

[12] Ramakrishnan M. and Sahaya Anselin Nisha A., "Groundnut Leaf Disease Detection and Classification by using Back Probagation Algorithm". IEEE International Conference on Communications and Signal Processing (ICCSP), Melmaruvathur 2015.

[13]Ahmad Nor Ikhwan Masazhar and Mahanijah Md Kamal, "Digital Image Processing Technique for Palm Oil Leaf Disease Detection using Multiclass SVM", IEEE 4th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA), Malaysia 2017.

[14] Aditya Parikh, Mehul S. Raval, Chandrasinh Parmar and Sanjay Chaudhry, "Disease Detection and Severity Estimation in Cotton Plant from Unconstrained Images", IEEE International Conference on Data Science and Advanced Analytic, Canada 2016.