# Plant Disease Detection Using Convolutional Neural Network

**Shaik Jameel**
Malla Reddy College of Engineering & Technology, India

*Abstract*— When plants and crops are affected by pests it affects the agricultural production of the country. Usually farmers or experts observe the plants with naked eye for detection and identification of disease. But this method can be time processing, expensive and inaccurate. Automatic detection using image processing techniques provide fast and accurate results. This paper is concerned with a new approach to the development of plant disease recognition model, based on leaf image classification, by the use of deep convolutional networks. Advances in computer vision present an opportunity to expand and enhance the practice of precise plant protection and extend the market of computer vision applications in the field of precision agriculture. Novel way of training and the methodology used facilitate a quick and easy system implementation in practice. All essential steps required for implementing this disease recognition model are fully described throughout the paper, starting from gathering images in order to create a database, assessed by agricultural experts, a deep learning framework to perform the deep CNN training. This method paper is a new approach in detecting plant diseases using the deep convolutional neural network trained and fine-tuned to fit accurately to the database of a plant's leaves that was gathered independently for diverse plant diseases. The advance and novelty of the developed model lie in its simplicity; healthy leaves and background images are in line with other classes, enabling the model to distinguish between diseased leaves and healthy ones or from the environment by using CNN.

***Keywords:*** Plant Disease Detection, Convolutional Neural Network

## I. INTRODUCTION

The problem of efficient plant disease protection is closely related to the problems of sustainable agriculture Inexperienced pesticide usage can cause the development of long-term resistance of the pathogens, severely reducing the ability to fight back. Timely and accurate diagnosis of plant diseases is one of the pillars of precision agriculture. It is crucial to prevent unnecessary waste of financial and other resources, thus achieving healthier production in this changing environment, appropriate and timely disease identification including early prevention has never been more important. There are several ways to detect plant pathologies. Some diseases do not have any visible symptoms, or the effect becomes noticeable too late to act, and in those situations, a sophisticated analysis is obligatory. However, most diseases generate some kind of manifestation in the visible spectrum, so the naked eye examination of a trained professional is the prime technique adopted in practice for plant disease detection. In order to achieve accurate plant disease diagnostics a plant pathologist should possess good observation skills so that one can identify characteristic symptoms. Variations in symptoms indicated by diseased plants may lead to an improper diagnosis since amateur gardeners and hobbyists could have more difficulties determining it than a professional plant pathologist. An automated system designed to help identify plant diseases by the plant's appearance and visual symptoms could be of great help to amateurs in the gardening process and also trained professionals as a verification system in disease diagnostics. Advances in computer vision present an opportunity to expand and enhance the practice of precise plant protection and extend the market of computer vision applications in the field of precision agriculture. Exploiting common digital image processing techniques such as color analysis and thresholding were used with the aim of detection and classification of plant diseases. In machine learning and cognitive science, ANN is an information-processing paradigm that was inspired by the way biological nervous systems, such as the brain, process information. Neural networks or connectionist systems are a computational approach used in computer science and other research disciplines, which is based on a large collection of neural units (artificial neurons), loosely mimicking the way a biological brain solves problems with large clusters of biological neurons connected by axons. Each neural unit is connected with many others, and links can be enforcing or inhibitory in their effect on the activation state of connected neural units. Each individual neural unit may have a summation function which combines the values of all its inputs together. There may be a threshold function or limiting function on each connection and on the unit itself, such that the signal must surpass the limit before propagating to other neurons. These systems are self-learning and trained, rather than explicitly programmed, and excel in areas where the solution or feature detection is difficult to express in a traditional computer program. Neural networks typically consist of multiple layers or a cube design, and the signal path traverses from front to back. Back propagation is the use of forward stimulation to reset weights on the "front" neural units and this is sometimes done in combination with training where the correct result is known. More modern networks are a bit more free flowing in terms of stimulation and inhibition with connections interacting in a much more chaotic and complex fashion. Dynamic neural networks are the most advanced, in that they dynamically can, based on rules, form new connections and even new neural units while disabling others.

The goal of the neural network is to solve problems in the same way that the human brain would, although several neural networks are more abstract. Modern neural network projects typically work with a few thousand to a few million neural units and millions of connections, which are still several orders of magnitude less complex than the human brain and closer to the computing power of a worm. New brain research often stimulates new patterns in neural networks. One new approach is using connections which span much further and link processing layers rather than always being localized to adjacent neurons. Other research

being explored with the different types of signal over time that axons propagate, such as Deep Learning, interpolates greater complexity than a set of Boolean variables being simply on or off. Their inputs can also take on any value between 0 and 1. Also, the neuron has weights for each input and an overall bias. The weights are real numbers expressing importance of the respective inputs to the output. The bias is used for controlling how easy the neuron is getting to output 1. For a neuron with really big bias it is easy to output 1, but when the bias is very negative then it is difficult to output 1.

## II. MATERIALS AND METHODS:

The Dataset: - The Dataset was taken from Kaggle of Plant Village dataset present online as such the code was also written on the online kernel of Kaggle for better computation and analysis of training loss and validation.

## III. IMAGE PREPROCESSING AND LABELLING

Preprocessing images commonly involves removing low-frequency background noise, normalizing the intensity of the individual particles images, removing reflections, and masking portions of images. Image preprocessing is the technique of enhancing data Furthermore, procedure of image preprocessing involved cropping of all the images manually, making the square around the leaves, in order to highlight the region of interest (plant leaves). During the phase of collecting the images for the dataset, images with smaller resolution and dimension less than 500 pixels were not considered as valid images for the dataset. In addition, only the images where the region of interest was in higher resolution were marked as eligible candidates for the dataset. In that way, it was ensured that images contain all the needed information for feature learning. Many resources can be found by searching across the Internet, but their relevance is often unreliable. In the interest of confirming the accuracy of classes in the dataset, initially grouped by a keywords search, agricultural experts examined leaf images and labeled all the images with appropriate disease acronym. As it is known, it is important to use accurately classified images for the training and validation dataset. Only in that way may an appropriate and reliable detecting model be developed. In this stage, duplicated images that were left after the initial iteration of gathering and grouping images into classes were removed from the dataset.

## IV. NEURAL NETWORK TRAINING

Training the deep convolutional neural network for making an image classification model from a dataset was proposed. Tensor Flow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. Tensor Flow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well. In machine learning, a convolutional neural network is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex. Individual cortical neurons respond to stimuli in a restricted region of space known as the receptive field. The receptive fields of different neurons partially overlap such that they tile the visual field. The response of an individual neuron to stimuli within its receptive field can be approximated mathematically by a convolution operation. Convolutional networks were inspired by biological processes and are variations of multilayer perceptron designed to use minimal amounts of pre-processing. They have wide applications in image and video recognition, recommender systems and natural language processing. Convolutional neural networks (CNNs) consist of multiple layers of receptive fields. These are small neuron collections which process portions of the input image. The outputs of these collections are then tiled so that their input regions overlap, to obtain a higher-resolution representation of the original image; this is repeated for every such layer. Tiling allows CNNs to tolerate translation of the input image. Convolutional networks may include local or global pooling layers, which combine the outputs of neuron clusters. They also consist of various combinations of convolutional and fully connected layers, with point wise nonlinearity applied at the end of or after each layer. A convolution operation on small regions of input is introduced to reduce the number of free parameters and improve generalization. One major advantage of convolutional networks is the use of shared weight in convolutional layers, which means that the same filter (weights bank) is used for each pixel in the layer; this both reduces memory footprint and improves performance. The layer's parameters are comprised of a set of learnable kernels which possess a small receptive field but extend through the full depth of the input volume. Rectified Linear Units (Re LU) are used as substitute for saturating nonlinearities. This activation function adaptively learns the parameters of rectifiers and improves accuracy at negligible extra computational cost. In the context of artificial neural networks, the rectifier is an activation function defined as:

$$f(x)=\max(0,x)$$

where x is the input to a neuron. This is also known as a ramp function and is analogous to half-wave rectification in electrical engineering. This activation function was first introduced to a dynamical network by Hahn loser et al. in a 2000 paper in Nature with strong biological motivations and mathematical justifications. It has been used in convolutional networks more effectively than the widely used logistic sigmoid (which is inspired by probability theory; see logistic regression) and its more practical counterpart, the hyperbolic tangent. The rectifier is, as of 2015, the most popular activation function for deep neural networks. Deep CNN with ReLUs trains several times faster. This method is applied to the output of every convolutional and fully connected layer. Despite the output, the input normalization is not required; it is applied after ReLU nonlinearity after the first and second convolutional

layer because it reduces top-1 and top-5 error rates. In CNN, neurons within a hidden layer are segmented into "feature maps." The neurons within a feature map share the same weight and bias. The neurons within the feature map search for the same feature. These neurons are unique since they are connected to different neurons in the lower layer. So for the first hidden layer, neurons within a feature map will be connected to different regions of the input image. The hidden layer is segmented into feature maps where each neuron in a feature map looks for the same feature but at different positions of the input image. Basically, the feature map is the result of applying convolution across an image. The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input and producing a 2-dimensional activation map of that filter. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input. Stacking the activation maps for all filters along the depth dimension forms the full output volume of the convolution layer. Every entry in the output volume can thus also be interpreted as an output of a neuron that looks at a small region in the input and shares parameters with neurons in the same activation map. When dealing with high-dimensional inputs such as images, it is impractical to connect neurons to all neurons in the previous volume because such network architecture does not take the spatial structure of the data into account. Convolutional networks exploit spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers: each neuron is connected to only a small region of the input volume. The extent of this connectivity is a hyper parameter called the receptive field of the neuron. The connections are local in space (along width and height), but always extend along the entire depth of the input volume. Such architecture ensures that the learnt filters produce the strongest response to a spatially local input pattern. Three hyper parameters control the size of the output volume of the convolutional layer: the depth, stride and zero-padding.

### *A. Depth*

Depth of the output volume controls the number of neurons in the layer that connect to the same region of the input volume. All of these neurons will learn to activate for different features in the input. For example, if the first Convolutional Layer takes the raw image as input, then different neurons along the depth dimension may activate in the presence of various oriented edges, or blobs of color.

### *B. Stride*

Stride controls how depth columns around the spatial dimensions (width and height) are allocated. When the stride is 1, a new depth column of neurons is allocated to spatial positions only 1 spatial unit apart. This leads to heavily overlapping receptive fields between the columns, and also to large output volumes. Conversely, if higher strides are used then the receptive fields will overlap less and the resulting output volume will have smaller dimensions spatially.

### *C. Stride controls*

Stride controls how depth columns around the spatial dimensions (width and height) are allocated. When the stride is 1, a new depth column of neurons is allocated to spatial positions only 1 spatial unit apart. This leads to heavily overlapping receptive fields between the columns, and also to large output volumes. Conversely, if higher strides are used then the receptive fields will overlap less and the resulting output volume will have smaller dimensions spatially.

Parameter sharing scheme is used in convolutional layers to control the number of free parameters. It relies on one reasonable assumption: That if one patch feature is useful to compute at some spatial position, then it should also be useful to compute at a different position. In other words, denoting a single 2-dimensional slice of depth as a depth slice, we constrain the neurons in each depth slice to use the same weights and bias. Since all neurons in a single depth slice are sharing the same parameterization, then the forward pass in each depth slice of the CONV layer can be computed as a convolution of the neuron's weights with the input volume (hence the name: convolutional layer). Therefore, it is common to refer to the sets of weights as a filter (or a kernel), which is convolved with the input. The result of this convolution is an activation map, and the set of activation maps for each different filter are stacked together along the depth dimension to produce the output volume. Parameter Sharing contributes to the translation invariance of the CNN architecture. It is important to notice that sometimes the parameter sharing assumption may not make sense. This is especially the case when the input images to a CNN have some specific centred structure, in which we expect completely different features to be learned on different spatial locations. One practical example is when the input is faces that have been centred in the image: we might expect different eye-specific or hair-specific features to be learned in different parts of the image. In that case it is common to relax the parameter sharing scheme, and instead simply call the layer a locally connected layer. Another important layer of CNNs is the pooling layer, which is a form of nonlinear down sampling.

Pooling operation gives the form of translation invariance; it operates independently on every depth slice of the input and resizes it spatially. Overlapping pooling is beneficially applied to lessen over fitting. Also in favour of reducing over fitting, a dropout layer is used in the first two fully connected layers. But the shortcoming of dropout is that it increases training time 2-3 times comparing to a standard neural network of the exact architecture. Bayesian optimization experiments also proved that ReLUs and dropout have synergy effects, which means that it is advantageous when they are used together. The advance of CNNs refers to their ability to learn rich mid-level image representations as opposed to hand-designed low-level features used in other image classification methods.

## V. RESULTS AND CONCLUSION

The Results presented in this section are related to training with the whole database containing both original and augmented images. As it is known that convolutional networks are able to learn features when trained on larger datasets, results achieved when trained with only original images will not be explored. After fine-tuning the parameters of the network, an overall accuracy of 96.77% was achieved. Furthermore, the trained model was tested on each class individually. Test was performed on every image from the validation set. As suggested by good practice principles, achieved results should be compared with some other results. In addition, there are still no commercial solutions on the market, except those dealing with plant species recognition based on the leaves images. In this paper, a approach of using deep learning method was explored in order to automatically classify and detect plant dise ases from leaf images. The complete procedure was described, respectively, from collecting the images used for training and validation to image pre-processing and augmentation and finally the procedure of training the deep CNN and fine-tuning. Different tests were performed in order to check the performance of newly created model. As the presented method has not been exploited, as far as we know, in the field of plant disease recognition, there was no comparison with related results, using the exact technique.

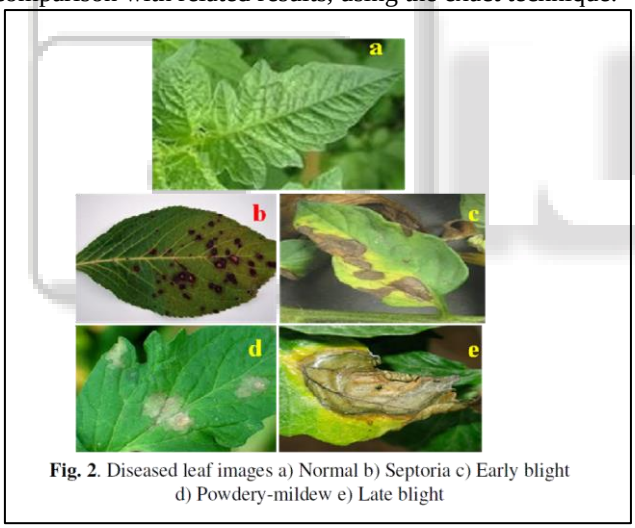
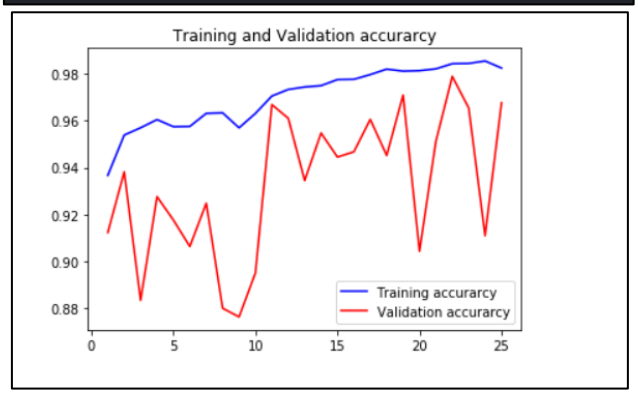
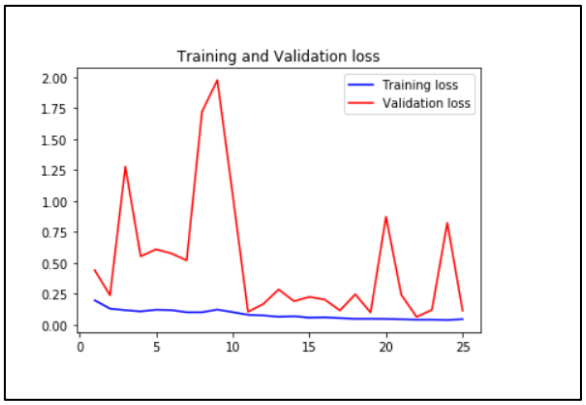



**Fig. 2**. Diseased leaf images a) Normal b) Septoria c) Early blight d) Powdery-mildew e) Late blight

```
[INFO] Calculating model accuracy
591/591 [==============================] - 2s 3ms/step
Test Accuracy: 96.77383080755192
```




## REFERENCES

[1] Aakanksha Rastogi, Ritika Arora, Shanu Sharma, "Leaf Disease Detection and Grading using Computer Vision Technology &Fuzzy Logic," presented at the 2nd International Conference on Signal Processing and Integrated Networks (SPIN), IEEE, 2015, pp. 500–505.

[2] Garima Tripathi, Jagruti Save, "AN IMAGE PROCESSING AND NEURAL NETWORK BASED APPROACH FOR DETECTION AND CLASSIFICATION OF PLANT LEAF DISEASES," Int. J. Comput. Eng. Technol. IJCET, vol. 6, no. 4, pp. 14–20, Apr. 2015.

[3] S. Arivazhagan, R. Newlin Shebiah, S. Ananthi, S. Vishnu Varthini, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features," Agric Eng Int CIGR J., vol. 15, no. 1, pp. 211–217, Mar. 2013.

[4] Prof. Sanjay B. Dhaygude, Mr.NitinP.Kumbhar, "Agricultural plant Leaf Disease Detection Using Image Processing"IJAREEIE, vol. 2(1),pp. 599-602, January 2013.

[5] K. Muthukannan, P. Latha, R. PonSelvi and P. Nisha, "CLASSIFICATION OF DISEASED PLANT LEAVES USING NEURAL NETWORK ALGORITHMS," ARPN J. Eng. Appl. Sci., vol. 10, no. 4, pp. 1913–1918, Mar. 2015.

[6] Md. Nazrul Islam, M.A. Kashem, MahmudaAkter and Md. JamilurRahman, "An Approach to Evaluate Classifiers for Automatic Disease Detection and Classification of Plant Leaf," presented at the International Conference on Electrical, Computer and Telecommunication Engineering, RUET, Rajshahi-6204, Bangladesh, 2012, pp. 626–629