Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.776894

COTTON PLANT DISEASE DETECTION AND CLASSIFICATION USING DEEP LEARNING ALGORITHM

Shofia Priyadharshini.D[1],Guda Sainath Reddy[2] ,Thaidulapati Hari Prasad[2], Krishnan.M[2].

[1]Assistant Professor,Department of Electronics and Communication Engineering,Vel Tech High tech Dr.Rangarjan Dr.Sakunthala Engineering College,Chennai,India.

[2]UG Scholar,Department of Electronics and Communication Engineering,Vel Tech High tech Dr.Rangarjan Dr.Sakunthala Engineering College,Chennai,India.

ABSTRACT

Human visual observation and inspection are the most common methods of diagnosing cotton plant diseases. In addition to taking a long time and requiring considerable effort on the part of the researcher, it requires specialized knowledge and an extensive amount of time to complete. As a result of recent advancements in computer vision and deep learning, a system for the detection of plant diseases may be developed in which disease can be detected across a wide geographical area with a minimum of intervention on the part of humans. A variety of neural network algorithms have been evaluated across a variety of experiments, such as Mini batch gradient descent and Faster RCNN. A total of 54305 images were used in all of the experiments, representing 38 different classes of plant diseases. The performances of each architecture were evaluated using four different metrics: accuracy, precision, recall, and F1 score. Compared with other convolutional neural network (CNN) architectures, our proposed model performed better than all other Convolutional Neural Network (CNN) architectures by an average of 95.12% and 96.98% respectively for 6 and 40 training epochs.

Keywords: *Cotton Plant Disease, Detection, Classification, Mini Batch Gradient Descent, Faster RCNN.*

1. INTRODUCTION

Diseases of cotton plants pose a substantially negative impact on food security, negatively affecting both quality and quantity of agricultural yields. Approximately 25% of global crop losses are caused by diseases, insect pests, or weeds. Across the United States, plant diseases cause an annual loss of around $40 billion in crop yields. The majority of developing countries, in particular, rely on smallholder farmers for food production; however, traditional methods of detecting plant diseases are both costly and time-consuming, making them unsuitable for smallholders. Deep Learning technologies have recently improved, and the large volume of collected data indicates that this is an effective way for achieving this goal in a cost-effective and efficient manner.

Various plant pathogens can cause cotton plant diseases, including fungi, bacteria, amoebae, fungal spores, nematodes, and parasitic organisms. The presence of plant diseases can have a detrimental impact on any plant system, particularly agricultural systems, which can adversely affect human livelihoods as well as health. Accordingly, late blight affecting potatoes, which is caused by Phytophthorainfestans-- an oomycete pathogen similar to fungi, was discovered in Ireland in the early 19th century. Following the outbreak of the epidemic, it is estimated that over 1 million people perished as a result of starvation, resulting in approximately 2 million

people emigrating abroad. Major agricultural disease outbreaks can have a profound impact on human health. Less massive outbreaks are more likely to result in reduced yields, a negative impact on the economy, and a particularly detrimental impact on small farmers and subsistence farmers.

There are many methods that are traditionally used in cotton plant disease detection and tracking. Visual inspection is one of the most common methods used. Sometimes, microscopic observation of molecular structures is helpful in the detection of diseases. The methods described in this section are generally accurate; however, their application is limited in spatial terms. It is necessary to develop a system that is accurate and affordable, which will enable farmers, in particular small farmers, early detection of plant diseases, over a wide range of fields, over a wide range of seasons.

2 RELATED WORKS

Sardogan, M., et al. in 2018 [1] proposed a framework for the diagnosis and classification of tomato plant leaf diseases using a mix of CNN and LVQ.Wallelign, S., et al. in 2018 [2] suggested the use of leaf images, we addressed the potential using CNN algorithms to categorize different plant diseases. The methodology indicated above is achieved by leveraging the LeNet, a common CNN architect in order to identify diseases in soybean plants

Sladojevic, S., et al. in 2016 [3] proposed about the development of a new-age algorithm for the diagnosis of 13 plant illnesses from healthy plants leaf images. Fuentes, A., et al. in 2017 [4] provided a framework that may be implemented in two steps. Initially, the SSD, R-FCN and Faster R-CNN meta-architectures integrated in to produce an unique meta-architecture.

Arivazhagan, S. and Ligi, S. V. in 2018 [5] developed a system focused on automating deep learning for disease identification and categorization in the mango plants. A collection of 1200 images of degraded and well-preserved mango leaves was used for developing this framework.Oppenheim, D. and Shani G. in 2017 [6] suggested recognizing and classifying different illnesses in potato plants in accordance with architecture of CNN. This technology makes use of a dataset of 2465 potato images. Barbedo, J. G. A. researched and found the advantages and drawbacks of several parameters that impact the model and performance of DL network used for the detection and analysis of several plant diseases in 2018 [7].

Brahimi, M., et al. in 2017 [8] suggested a system based on a CNN for detecting and classifying different tomato harvest illnesses. The collection of the architecture comprises of 14,828 tomatoes leaf images from the plants village image collection with nearly nine illnesses. The proposed methodology has an efficiency of 98.18%. Shrivastava, V. K., et al. [9] focussed on the recognition and organization of different illnesses among rice plants utilizing a outline supported by CNN architecture and SVM in 2019.

Ozguven, M. M. and Adem, K. in 2019 [10] In the instance of sugar beet, the remaining quicker region-based CNN methodology was modified by altering the criteria for identifying disease-affected regions. The dataset contains 155 sugar beet images; suggested outline was acheived in the accuracy ratio of 94.58%. In 2020 [11], Uguz, S., and Uysal, N. proposed CNN architectures, VGG16 and VGG19 architectures, were evaluated in a tranferable learning scenario in the situation of Olive plant illnesses.

Agarwal, M. et al. 2020 [12] A modified approach based on CNN detection of tomato leaf disease was proposed. In addition, the suggested model was compared to ML models and VGG-16. As a result of our suggested model of 98% accuracy, the KNN model obtained 95%

accuracy and the VGG16 model yielded 94% accuracy. This framework's tomato leaf pictures dataset is derived from the Plant village dataset. Wang, J. et al. [13] investigated a transferring scenario for learning, depending on CNN structure for disease identification and organization using leaf pictures from two crops, cucumbers and rice, in 2018.

Toda, Y., and Okura, F. in 2019 [14] suggested the influence of DL technology means of leaf images was discussed. The architecture of CNNpurposes a black box typical for plant disease diagnostics. The many components of hyperparameters that impact classifying accuracy are also covered. Using deep learning scenarios, many prototypes and framework have been established in terms of recognition and categorization of distinct categorical illnesses in the specific plant. Deep learning may also be used to identify and analyse the macronutrient composition of a particular plant. For instance, Tran, T. T. et al. in 2019 [15] developed a strategy based on a DL scenario that provides a observing platform that maintains a monitoring system to multiple phases upon germination to producing to obtain a higher rate of harvest. In order to construct the proposed system, 571 images of tomato leaves and fruits images at different phases of crop development.

According to EhsanKiani et.al, 2017 [16] proposed a image segmentation using colors, i.e. colour image segmentation approaches, aids in better understanding and resolution of the problem.

3 PROPOSED METHODOLOGY

This paper contributes primarily to the following areas. Proposing RPN is a designed proposal whose scale and ratio can be varied. With RPN, object detection is guided to look by implementing neural network terminology.Instead of using image pyramids or filter pyramids, anchor boxes were introduced in this paper. As the name implies, anchor boxes are reference boxes with fixed dimensions and aspect ratios. As a result of the use of several reference anchors, the region is capable of displaying a variety of scales and aspects. The reference anchor boxes may be viewed as a pyramid. A mapping is then performed between each region and an anchor box.A convolutional computation is shared between the R-CNNand the Fast RPN. As a result, computational time is reduced.Combined FRCNN with Mini Batch Gradient Descent method to get more accurate results.

3.1 Mini-batch Gradient Descent

The Gradient Descent algorithm computes a single gradient step for the entire batch dataset and updates the parameter vector accordingly. As it has immediate access to the entire training set, this method is more accurate. However, it may become extremely computationally costly as the training size increases to millions or billions of examples. In addition, this may lead to underutilization of the parallel computing capabilities provided by contemporary multicore architectures. Due to this, instead of computing the gradient from the entire dataset or from a single data point, a small sample of data is more often used in mini-batch gradient descent as it can reduce training time while maintaining convergence. Gradient descent is performed in batches of m′ samples using the mini-batch algorithm.

Algorithm: Mini-Batch Gradient Descent Algorithm

Step 1: Random initialization of weights

Step 2: Loop upto convergence

Step 3: *for (i=1 to size-of-data) do:*
Step 4: *for (j = 1 to m) do:*
Step 5: *mini-Batch Gradient computation*
$j(\theta) = i(\theta) + data\_size$
Step 6: $j(\theta) = \frac{1}{data\_size} \times j(\theta)$
Step 7: update overall weight,
Step 8: Return overall weight

Moreover, a large batch size (more than 128) is detrimental in terms of the generalizability of the model. According to my experience, starting with a size of batch 1 and maintaining low initial rates, then gradually increasing the batch size (e.g., 8, 16, 32), is the best way to reduce training time and maintain a high level of generalization.

3.2 Faster RCNN

The system consists of two modules: **RPN**: In order to generate proposals for regions.**Fast R-CNN**: The purpose of this is to detect objects within the regions proposed.It is the responsibility of the RPN module to generate region proposals. A neural network is based on the principle of attention, which indicates which objects should be searched for an image in the Fast R-CNN detection module as shown in fig 1.

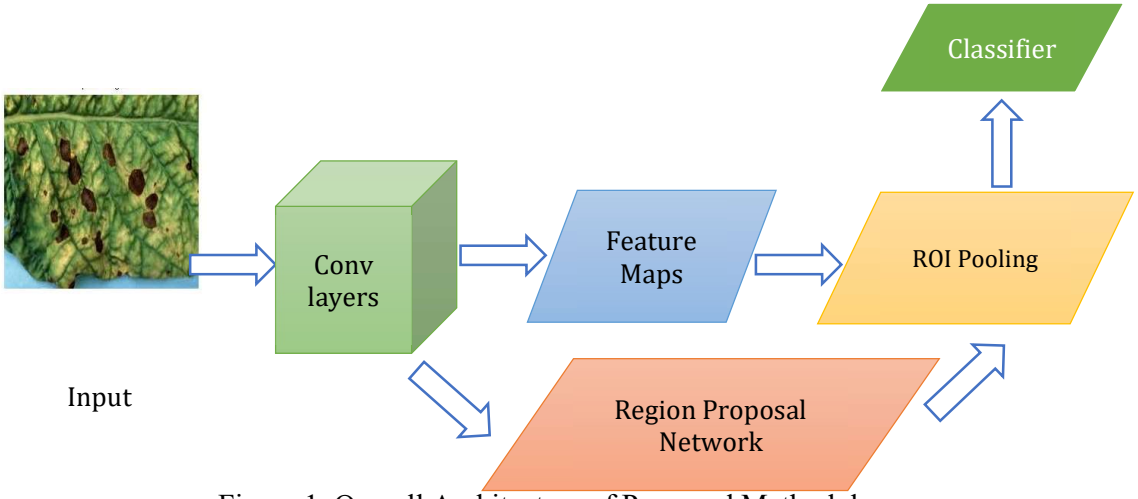

Figure 1. Overall Architecture of Proposed Methodology

It is important to note that both Fast R-CNN and RPN share convolutional layers.

As a result of the Faster R-CNN, the following occurs:

- Proposals are generated by the RPN.
- The ROI Pooling layer is used to extract fixed-length feature vectors for the regions proposed in the image.
- A Fast R-CNN is used to classify the extracted feature vectors.
- A class score and bounding box are returned for each detected object.

3.2.1 Region Proposal Network (RPN)

A Selective Search algorithm is used in both the Fast R-CNN and R-CNN models to generate region proposals. A pre-trained CNN is used to classify each proposal. The paper proposes a network called region proposal network (RPN) for producing region proposals. There are several advantages to this approach:

1. A network is now used to generate region proposals according to the detection task. It is possible to train the network according to the nature of the detection task and customized accordingly.
2. Based on the use of a network to generate the proposals, a customized detection algorithm can be developed using this network end-to-end. As a result, it produces more accurate region proposals than generic methods such as select search and edge boxes.
3. Similarly to the Fast R-CNN detection network, the comparisons to algorithms such as Selective Search, RPN does not require additional time to generate proposals.
4. It is possible to merge/unify Fast R-CNN and RPN.Since they share the same convolutional layers. As a result, there is only one training session.

The feature map is traversed by a sliding window which is based upon a rectangular window of size nxn. It is possible to generate several proposals for candidate regions for each window. This is not the final version of the proposal, as it will be evaluated according to its "objectivity score".

3.2.2 Anchor

In the figure below, the last convolutional layer feature map passes within a rectangular sliding region of size nxn, with n equal to 3 in the case of the VGG-16 network. We generate proposals for K regions for each window. A reference box known as an anchor box is used to parametrize each proposal. There are two parameters associated with the anchor boxes:

1. Scale
2. Aspect Ratio

As a general rule, there are three scales and three aspect ratios, so there are K=9 anchor boxes in total. However, K may differ from 9. As a result of each region proposal, K regions are generated, with each region having a different scale or aspect ratio. The following figure 2 illustrates some of the anchor variations.

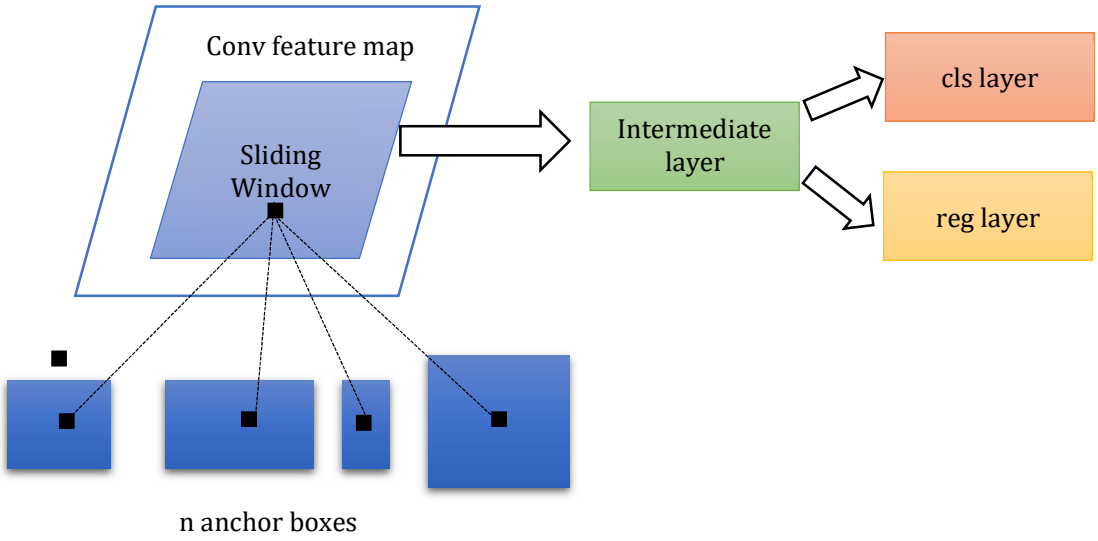

Figure 2. Anchor Structure

In order to detect objects with scale-invariant detection, a single image at one scale must be used to detect objects with scale-invariant detection, since the anchors vary in scale. As a result, multiple images or filters are avoided.

In the case of each proposed NXN region, a vector of features is obtained. Following this, two fully-connected layers are fed with this vector:

1. First, the FC layer is called cls, which is an algorithm that calculates regional objectness scores.
2. Second FC layer is called as reg, the bounding box of the region is defined by a four-dimensional vector.

There are two outputs on the first FC layer. Firstly, the region is classified as a background, and then it is classified as an object. Objectness scores for each anchor are assigned and used to produce classification labels in the following section.

3.2.3 Pooling Layer

An image can be compressed using the pooling layer by substituting a summary statistic for the feature map output. Similarly, in max-pooling, a sliding window is calculated using the maximum statistic, which downscales the feature map accordingly, reducing memory consumption and speeding up training. A sliding window will be computed with the average statistic in average pooling, resulting in an averaged feature map. When a deep neural network is built with a pooling layer, images can carry less information.

3.2.4 Activation Functions

It should be noted that deep neural networks differ from the preliminary multilayerperceptrons that use linear activation functions, in that they are trained in the form of linear operations stacked upon each other. Based on Cover's theorem: “It is more likely that complex classification problems can be linearly separated in high-dimensional nonlinear spaces than in low-dimensional input spaces”. We are providing an overview of this topic in this sectionseveral activation functions that are commonly used in deep learning studies, such asReLU (rectified linear unit), softmax, hyperbolic tangent and sigmoid and their variances.

3.2.5 Softmax

SoftMax's equation is as follows:

$$pi = g(zi) = Softmax(zi) = \frac{e^{zi}}{\sum_{j=1}^{n} e^{zj}}$$

wherezi is the resulting probability depends on the linear combiner, and pi represents the probability generated for class i. For example, the following is an expression of a linear combiner:

$$zj = \sum_{I=1}^{M} W_{ij}xi + bias$$

where i represents the neuron in the first layer, and j represents the neuron in the second layer. Multiclass classification tasks often require the use of the SoftMax function, which compresses real numbers as low as 0 and as high as 1 to ensure they sum to 1:

$$\sum_{i=1}^{m} pi = 1$$

In the following example, one can see how this property is manifested in the application of softmax to the classification of multi-class neural networks.

In comparison with sigmoid and tanh, ReLU is a more economical operation, especially when training deep neural networks since it prevents gradient vanishing. According to some research, a rectified neural network provides an absolute error rate deduction of 2% compared to a sigmoidal unit [29]. ReLU can be calculated using the following equation:

$$g(z) = Relu(z) = max(0, z)$$

A ReLU function's first derivative is: $g'(z) = \begin{cases} 1, & z > 0) \\ 0, & otherwise \end{cases}$

3.3 Objectness Score

Each region proposal is output as a vector with two elements by the CLS layer. When element 1 is 1 and element 2 is 0, the proposal for the region is considered to be background.

The area of union is calculated in proportion to the intersection area and the area of union of the anchor box. As the two boxes become closer to one another, the IoU increases until it reaches 1.0 (when both boxes are 100% similar).

Objectness scores are assigned to anchors under the following four conditions based on the IoU:

1. Positive objectness labels are assigned to anchors with an IoU overlap of greater than 0.7 when compared to an actual ground truth box.
2. Positive labels may be assigned to those anchors with the highest IoU overlap if there are no anchors with IoU overlaps higher than 0.7.
3. In the event that the overlap of IoUs between all truth boxes equals or is less than 0.3, a non-positive anchor is consigned a negative objectivity score. When the anchor's objectivity score is negative, background is assigned to the anchor.
4. If the anchor does not contribute to the training objective, it is not considered an anchor.

$$Objectness\ Score\ (IoU) = \begin{cases} Positive\ \rightarrow IoU > 0.7 \\ Positive\ \rightarrow 0.5 < IoU\ \leq 0.7 \\ Negative\ \rightarrow IoU < 0.3 \\ Not\ NegativePositive\ \rightarrow 0.3\ \leq IoU\ \leq 0.5 \end{cases}$$

Normally, the first assumption (0.7 <IoU) suffices for anchors to be classified as positive, however the authors included a second assumption (0.5 <IoU<= 0.7) in the event that no regions have an IoU of 0.7.

4 RESULTS AND DISCUSSIONS

4.1 Training

In the present study, the originating PlantVillage datasets have been separated into validation and training datasets according to 80% and 20%, correspondingly, in accordance with Table 3. Validation data is used for evaluating performance after each epoch and not used in training. Before training, the pixels of the images were first normalized by dividing them by 255, all models were run with an input size of 256x256 in the default mode, and utilizing batch sizes of 32, as this provided the highest level of accuracy.

4.2 Evaluation Metrics

Evaluating a dataset with an unbalanced dataset requires the selection of appropriate evaluation metrics as well as the selection of a learning algorithm. Following each epoch of training, the following statistics were collected: time per epoch, cross-entropy losses, reliability, accuracy, precision, and Area Under the Curve(AUC).

4.3 Accuracy

Accuracy is the indicator that has the widest application and it is the simplest in estimating whether or not correct calculations were made over a dataset.The definition of accuracy as a percentage of predictions that have been corrected is described below: [14]:

$$Accuracy = \frac{\#\ of\ corected\ prediction}{Total\ samples}$$

4.4 Precision

According to the definition of precision, it refers to the percentage of true positives based on the total number of positives retrieved. As a general rule, precision lies in the range of 0 to 1, as shown in the following equation:

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{Total\ Predicted\ Positive}$$

4.5 Recall

Recall is calculated as the number of true positive examples multiplied by the total number of TP examples. This is often referred to as the True Positive Rate (TPR). It is also possible to calculate the recall range using the following equation:

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{Total\ Actual\ \ Positive}$$

4.6 F1 Score

F1 scores have been defined as the weighted harmonic average of precision and recall, which has been widely incorporated into ML algorithms. This equation can be expressed as follows:

$$F1 = 2\ \times \frac{Precision * Recall}{Precision + Recall}$$

In the development of deep learning, we were unable to determine the parameters that would produce the most accurate results at the outset. As a result, applying DL is a process that requires repetition, since we need to go through the entire process of developing the idea, developing the model, assessing its performance, repeatedly. As a result, one of the ways to accelerate the process of development involves improving the efficiency of the development cycle. This involves establishing the proper partitioning for training, evaluation, and implementation.

A total of seven experiments were conducted to determine the most appropriate number of batches for this study.

4.7 Loss Function

The PlantVillage dataset contains 38 classes, hence two commonly used categorical loss functions were selected as loss functions for this study:

(i) Categorical CrossEntropy (CCE) and (ii) Sparse Categorical Cross-Entropy (SCCE).

In multiclass classification, CCE is the most commonly used loss function, as shown below:

$$Generalized\ cross - entropy = -\sum_{i}^{c} yi \log(xi;\ \theta)$$

Where $\boldsymbol{yi}$ and fi $(\boldsymbol{xi})$, represents the encoded labels and DNN values for each $classi$ in C. Target scores were computed using one-hot encoding. An accuracy of training after the first epoch using various batch sizes as given in table 1.

Table 1: Accuracy of training after the first epoch using various batch sizes

Size of Batch	Accuracy of Valid	Accuracy of Train	Time / Epoch
32	0.97	0.92	1062
64	0.84	0.92	1158
128	0.92	0.93	1086
256	0.74	0.91	1262
512	0.61	0.89	1048

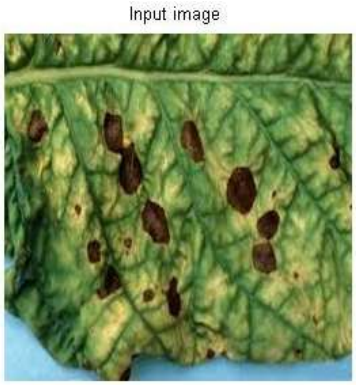

Figure 3. Input Image of cotton leaf

Figure 4. Preporcessing

Figure 5. Morphological Processing

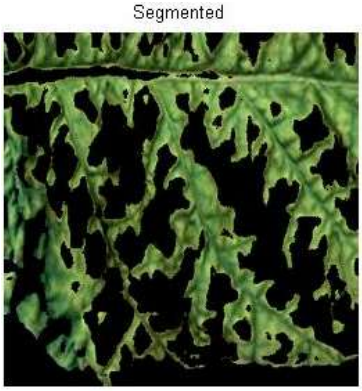

Figure 6. Image Segmentation

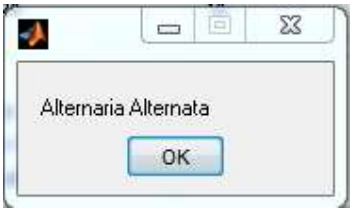

Figure 7. Image Classification

From the fig 3-7 shows the results of cotton leaf disease prediction and classification with accuracy of 96.98% which is better than existing methods.

Conclusion

As the main objective of this approach, it is to identify and classify diseases in cotton Leaves using FRCNN and including Mini batch gradient descent. Growth and yield of crops have a profound influence on the field of agriculture and on the farmer on every level, economically, socially, and in every way. To identify diseases at the right time, it is necessary to monitor

crops carefully at various stages of growth. To ensure accurate identification of diseases of a specific crop, automatic recognition and classification are required. Traditionally, cotton plant diseases are diagnosed by humans by observing and examining them visually. This approach, however, requires a great deal of time and expertise. Recent advancements in computer vision and deep learning, a system for detecting plant diseases across large areas could be developed without requiring a great deal of human intervention. According to this study, a DL approach was developed for classifying plant leaf diseases. Experiments were conducted on a variety of neural network architectures to quantify their performance, which included the Mini batch gradient descent algorithm and the Faster RCNN. There are a total of 54305 images in the PlantVillage dataset that cover a wide range of plant diseases. All experiments were trained on this dataset. The performance of each architecture was evaluated based on four different metrics: accuracy, precision, recall, and F1 score. Compared to all other CNN architectures, our model achieved training accuracy levels of 95.12% and 96.98% respectively for 6 and 40 training epochs.

References

1. Atila, Ü.,Uçar, M., Akyol, K., &Uçar, E. (2021). Plant leaf disease classification using EfficientNet deep learning model. *Ecological Informatics*, *61*, 101182.
2. Lu, J., Tan, L., & Jiang, H. (2021). Review on convolutional neural network (CNN) applied to plant leaf disease classification. *Agriculture*, *11*(8), 707.
3. Azim, M. A., Islam, M. K., Rahman, M. M., &Jahan, F. (2021). An effective feature extraction method for rice leaf disease classification. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *19*(2), 463-470.
4. Wang, C., Du, P., Wu, H., Li, J., Zhao, C., & Zhu, H. (2021). A cucumber leaf disease severity classification method based on the fusion of DeepLabV3+ and U-Net. *Computers and Electronics in Agriculture*, *189*, 106373.
5. Thangaraj, R., Anandamurugan, S., &Kaliappan, V. K. (2021). Automated tomato leaf disease classification using transfer learning-based deep convolution neural network. *Journal of Plant Diseases and Protection*, *128*(1), 73-86.
6. Sujatha, R., Chatterjee, J. M., Jhanjhi, N. Z., &Brohi, S. N. (2021). Performance of deep learning vs machine learning in plant leaf disease detection. *Microprocessors and Microsystems*, *80*, 103615.
7. Nandhini, S., &Ashokkumar, K. (2021). Improved crossover based monarch butterfly optimization for tomato leaf disease classification using convolutional neural network. *Multimedia Tools and Applications*, *80*(12), 18583-18610.
8. Chowdhury, M. E., Rahman, T., Khandakar, A., Ayari, M. A., Khan, A. U., Khan, M. S., ...& Ali, S. H. M. (2021). Automatic and reliable leaf disease detection using deep learning techniques. *AgriEngineering*, *3*(2), 294-312.
9. Chouhan, S. S., Singh, U. P., Sharma, U., & Jain, S. (2021). Leaf disease segmentation and classification of JatrophaCurcas L. and PongamiaPinnata L. biofuel plants using computer vision based approaches. *Measurement*, *171*, 108796.
10. Khalifa, N. E. M., Taha, M. H. N., El-Maged, A., Lobna, M., &Hassanien, A. E. (2021). Artificial Intelligence in Potato Leaf Disease Classification: A Deep Learning Approach. In *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges* (pp. 63-79). Springer, Cham.

11. Kaur, N. (2021). Plant leaf disease detection using ensemble classification and feature extraction. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, *12*(11), 2339-2352.
12. Applalanaidu, M. V., &Kumaravelan, G. (2021, February). A review of machine learning approaches in plant leaf disease detection and classification. In *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (pp. 716-724). IEEE.
13. Zhang, K., Wu, Q., & Chen, Y. (2021). Detecting soybean leaf disease from synthetic image using multi-feature fusion faster R-CNN. *Computers and Electronics in Agriculture*, *183*, 106064.
14. Sembiring, A., Away, Y., Arnia, F., &Muharar, R. (2021, March). Development of concise convolutional neural network for tomato plant disease classification based on leaf images. In *Journal of Physics: Conference Series* (Vol. 1845, No. 1, p. 012009). IOP Publishing.
15. Zhao, S., Peng, Y., Liu, J., & Wu, S. (2021). Tomato leaf disease diagnosis based on improved convolution neural network by attention module. *Agriculture*, *11*(7), 651.