International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Convolutional Neural Network with Extreme Learning for the Classification of Plant Leaf Diseases from Images

Swapna Jamal[1*] and John Edwin Judith[2]

[1*]*Department of Computer Applications, Noorul Islam Centre for Higher Education, Kumaracoil, Tamilnadu, India.*

[2]*Department of Computer Science and Engineering, Noorul Islam Centre for Higher Education, Kumaracoil, Tamilnadu, India*

E-mail address: swapnashan@gmail.com, judith@niuniv.com

ABSTRACT

Farmers are facing many difficulties right from the selection of seed to fertilizer usage, disease control, harvesting and selling the agricultural yield. The prime motivation behind this research stems from the idea that, the ability to detect leaf issues and implement corrective measures can offer a solution to mitigate the decrease in crop productivity. The existing Deep Learning methods like Convolutional Neural Network showed high efficiency regarding the modification and use of acquired knowledge. A novel framework has been developed by incorporating Convolutional Neural Network and tuning the hyperparameters. Training has been performed using Extreme Learning process which yielded better results. Convolutional Neural Network - Extreme Learning Algorithm is the underlying algorithm. The empirical study makes use of the Plant Village dataset. The leaf disease categories considered in this research early blight, black rot, bacterial spot, apple scab, cercospora leaf spot and healthy. Convolutional Neural Network - Extreme Learning achieved 94.28% precision, 95.63% accuracy, 94.68% recall, and 96.23% F1-score using Plant Village dataset, outperforming other classifiers. The research outcomes reflect that the proposed Deep Learning model and algorithm can be used real world computer vision applications pertaining to agriculture.

Keywords: Convolutional Neural Networks, Deep Learning, Extreme Learning, Hyperparameters, Plant Leaf Disease Classification.

E-mail:author's email

1. INTRODUCTION

Agriculture is the field that substantially contributes to the food economic growth of India. With technology innovations, it is time to incorporate precision agriculture in the country to bring in its benefits, such as reducing expenditure and increasing productivity [1]. When agricultural fields are monitored automatically with technology-driven solutions, it will change the way diseases are identified and counter-measures are taken from time to time [2]. When there is live monitoring and automatic detection of plant diseases with Artificial

Intelligence (AI) systems, it will improve agriculture with plenty of benefits. It will make agriculture and all its eco-systems healthy and prospering. Deep Learning (DL) techniques in Al have potential towards useful agricultural applications. One such useful application to farmer community is automatic detection of plant diseases [3]. It has attracted researchers of late due to availability of computing and storage resources besides advancements in Machine Learning (ML).

With advancements in Al-based methods such as DL, there is a need for a comprehensive DL framework that exploits advanced CNN models.

In agriculture, farmers used to observe crop leaves and identify certain diseases. However, the problem with the traditional approach is that, it becomes too late when a farmer identifies a disease and takes corrective measures. In other words, it does mean that there is a time gap between the inception of disease and clear identification of disease leading to damage to crop. This challenge can be overcome with technology-driven approaches that are, of late, given high importance in academia and research circles. Leaf disease detection with ML and DL has its own challenges. It is important to understand that a technology-driven approach needs to capture leaf images from remote places. The first challenge is the acquisition of leaves or the crop of a given farmer or location [4]. Once the cropped image is acquired, it is important to see that it has quality. Therefore, another challenge is to have acceptable quality of image in order to process it further. When there are plenty of crops in the world, there is need for large volumes of training data in order to have meaningful means of using DL. If there is no sufficient training data, it leads to deteriorated performance with DL algorithms.

When live plant images are captured remotely through certain technologies or sensors, there is a challenge involved in improving the quality of the input image. Leaf disease detection particularly throws challenges in terms of identification of the region of interest. It is also important to have training data for all pre-defined classes; otherwise, the detection performance proves to be average. With digital image processing there are certain issues such as segmentation and accuracy in processing [5]. There is a need for dynamically adaptive processes in order to process image data. Machine interpretation of image data is non-trivial that needs a high level of understanding from the underlying algorithms. With DL, the process of disease identification has its complexities besides the advantages aforementioned.

The rise of DL models in AI-driven computer vision applications opens up significant possibilities for automated identification of leaf diseases in farming. By implementing technology-driven solutions for automated monitoring of agricultural fields, the approach to disease identification and response mechanisms can undergo a transformative shift. Live monitoring and AI-driven detection of plant diseases promise many advantages for agriculture, ultimately contributing to the overall health and prosperity of agricultural ecosystems. The adoption of automatic disease detection through technological innovations is poised to make a profound impact on numerous nations worldwide. This shift will facilitate the adoption of precision agriculture practices, resulting in increased crop yields and reduced waste through timely and precise interventions. The motivation for this research stems from the potential far-reaching positive effects on agriculture.

2. RELATED WORKS

Ferentinos *et al*. [6] introduced CNN-based approach for efficient detection of plant diseases. It has been found that, for image-based inputs, CNN is able to perform good in ascertaining image features and detection. DeChant *et al.* [7] extended their research and included diversified crops and diseases. With the Plant Village dataset, their observations showed suitability and feasibility of using CNN for leaf disease detection. Yadav *et al.* [8] stated a DL model that is compatible for mobile devices. Since the usage of smart phones is found in all areas of life, their research is useful for mobile users as well. However, the applications in such devices depend on resources in the cloud investigated on the tomato crop, which is a widely grown vegetable crop in many countries. Durmus *et al.* [9] explored CNN-based DL models like SqeezeNet along with AlexNet for diagnosis of disease. The main investigation revealed that CNN-based pre-trained models have potential to help in improved leaf disease detection performance.

DL detection models such as Recurrent Neural Network (RNN) and CNN were studied by Voulodimos *et al.* [10] using different cases in agriculture. Particularly, they were good for diagnosis of leaf disease of different crops. Their observation was that, there is a capability with DL models to work on enormous amounts of training data in order to leverage detection performance significantly. They developed models based on CNN and found that image-related data is well learned by CNN for better detection and classification. Their empirical study using maize crops revealed its usefulness in further investigations. They also found CNN more suitable with different configurations such as dropouts and other parameters. Their observations showed the high utility of CNN in processing image inputs in the detection process of leaf disease. Tiwari *et al.* [11] studied different models of DL, including pre-trained ones such as VGG16. They suggested that CNN-based models have the potential to learn comprehensively while training and improve performance in the detection process. Gandhi *et al.* [12] found the significance of using CNN but in a network, model known as Generative Adversarial Network (GAN). This model with underlying CNN usage as a generator or discriminator leads to more useful discrimination. The reason is that it has the capacity to have data augmentation and improve quality in the training process and game theoretical approach that allows new training samples to be created to augment any data insufficiency.

Karthik *et al.* [13] used residual approach embedded in CNN model for working on tomato diseases. They configured the model with different layers and dropouts for better performance. The model was capable of finding diseases in the crop with high accuracy and intended to extend their models further to incorporate fault detection. Zhong *et al.* [14] investigated the model known as DenseNet-121 to detect diseases in apple crops. Their approach in detection needed multi-label classification that will have a number of class labels that are used in the testing phase. The disease identification in their research is not based on binary classification but deals with a multi-label approach to reflect true disease dynamics in the crop. Sullca *et al.* [15] proposed a enhanced LeNet model that is based on CNN baseline in order to have better performance in detection of diseases in agricultural crops. They used maize crop for their research and their model contains many layers for handling input data. From the state of the art, it is ascertained that there is a need for improving CNN-based models besides exploiting transfer learning, pre-trained models.

However, the existing work lacks in its optimal configuration and reuse of previously trained models. It also has shortcoming in terms of accuracy. There is need for further research on leaf disease detection using advanced CNN models with hyperparameter tuning and efficient learning technique. Therefore, this research is aimed at building a comprehensive model to address these limitations.

3. METHODOLOGY

This section explores the particular methods and approaches used in each part of the suggested categorization model. A key factor in the diversity and caliber of the dataset is image acquisition, which is the first phase of data collection. After that, preprocessing techniques like augmentation and scaling are used to make sure the dataset is ready to be prepared for analysis using CNN. Partitioning the dataset is an essential stage that permits the partition of the data into subsets for testing, validation, and training, guaranteeing thorough model analysis and generalization. The methodology's fundamental component is CNN, which works in tandem with Extreme Learning (CNN-EL) to extract complex features and patterns from the pre-processed data. Lastly, performance evaluation is essential for determining the model's effectiveness and correctness as well as its applicability for practical uses. These parts flow naturally together inside the block diagram, guaranteeing a methodical and comprehensive approach to picture analysis, classification, and assessment. This prepares the reader for the in-depth examination of each methodology component that follows. The block diagram for the suggested model is shown in Figure 1.

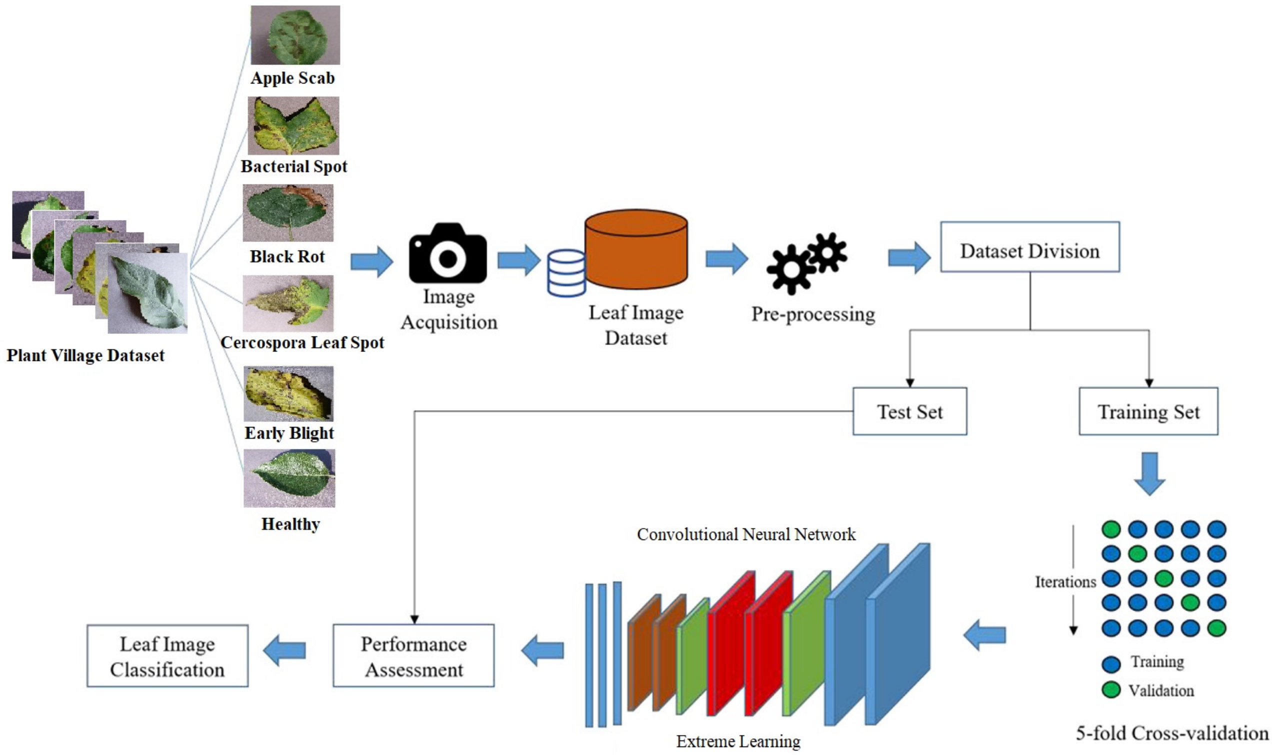


Figure 1: CNN-EL Model Block Diagram for Leaf Disease Classification

The proposed approach applies two distinct pre-processing techniques to pictures from the plant dataset. Using the image augmentation strategy increases the amount of images in the collection. More photos are required for Deep Neural Networks in order to improve training and validation accuracy. The augmentation process involves the application of several image processing techniques, such as flipping, cropping and rotating. By employing these techniques, the model's ability to recognize various iterations of an object is improved and the dataset becomes more varied. Resizing images is done with the intention of reducing the computational complexity associated with the DL process. Processing more pixels at once is necessary for large-sized images, which adds to the computational complexity and duration. at combat this, the sizes of the leaf images are standardized at 256x256, which increases computing efficiency without sacrificing the important substance of the photos. By reducing the data, the deep learning model can handle it more effectively without sacrificing its capacity to extract significant attributes. Figure 2 shows some photos taken from the Plant Village dataset.

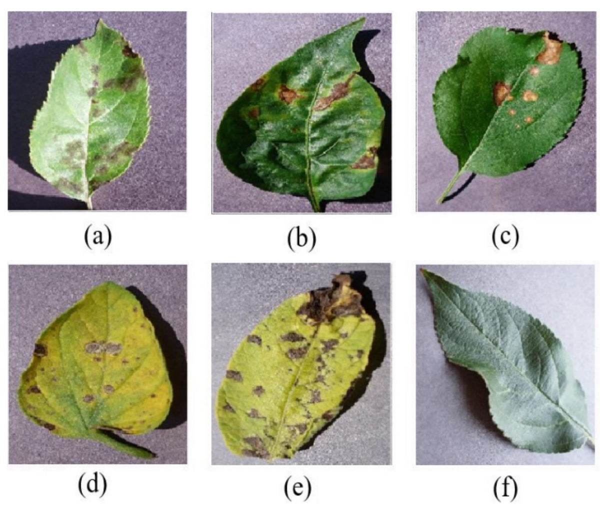


Figure 2: Sample Images from the plant village dataset: (a) apple scab, (b) spot of bacteria, (c) black rot, (d) Cercospora leaf spot, (e) early blight (f) healthy

Partitioning the dataset is a crucial stage in creating and assessing DL models. It involves dividing a dataset into discrete subsets with a purpose. Typically, these subsets are training and testing sets, but more sophisticated approaches may additionally include a validation set. The training set serves as the foundation for teaching the model, allowing it to identify and comprehend relationships, patterns, and features in the images [16]. In contrast, the testing set offers an objective standard by which to measure the model's effectiveness. A model's ability to extrapolate to new, untested data is crucial for ensuring that it can generate correct predictions in real-world applications and is not just a memory of the training images, which is why the data division process is so important [17]. Building strong, dependable, and efficient machine learning models requires proper dataset partitioning since it prevents problems like overfitting and makes it easier to conduct a thorough evaluation of a model's prediction power. In this case, the suggested ighty percent of the data is utilized for training and twenty percent of the data is used to test the CNN-EL model. Table 1 explains the distribution of the different image classes in the dataset.

Table 1: Images in Plant Village Dataset

Sl. No	Image Class	Total	Train	Test
1	Apple Scab	1260	1020	240
2	Bacterial Spot	1207	966	241
3	Black Rot	1242	990	252
4	Cercospora Leaf Spot	1260	1010	250
5	Early Blight	1200	959	241
6	Healthy	1240	982	258

A. *Convolutional Neural Network*

Convolutional Neural Network (CNN), is a powerful DL framework that can automatically extract meaningful information from data without requiring human intervention. CNNs excel in pattern recognition tasks, especially in images for identifying objects, people, and scenes. These networks consist of tens or even hundreds of layers, each playing a unique role in identifying various aspects of an image [18]. Every training image, regardless of its resolution, undergoes a series of filtering operations, and the output from each transformed image becomes the input for the subsequent layer. To capture intricate object characteristics, these filters often start with basic attributes like light and edges [19]. The basic structure of CNN used in leaf image classification application is illustrated in Figure 3.

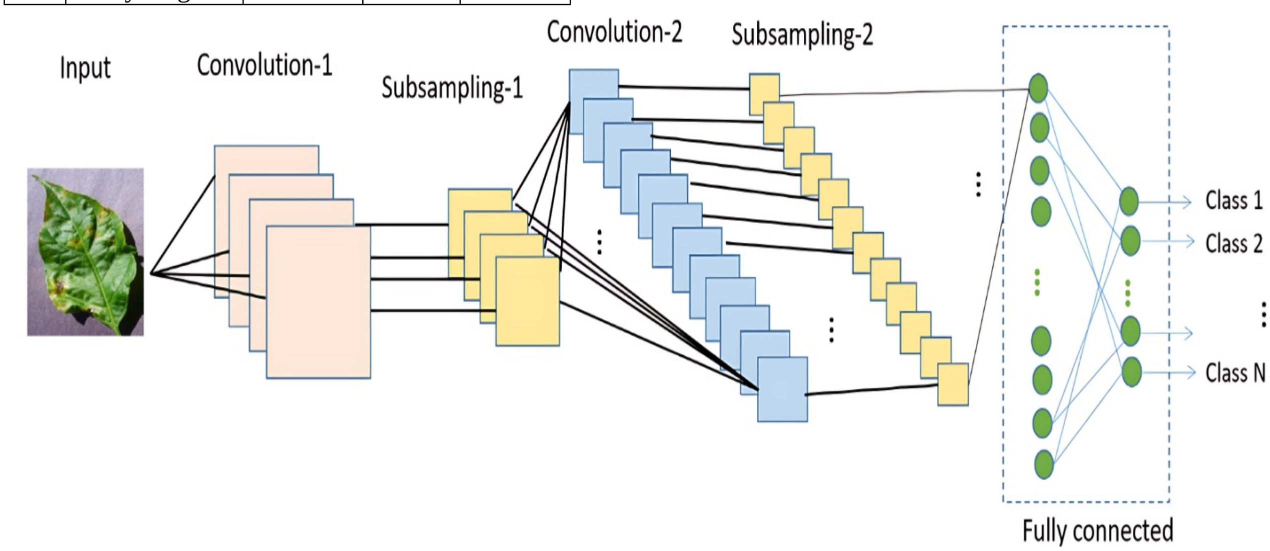


Figure 3: Basic Structure of CNN for Leaf Image Classification

CNN comprises many key components, containing several hidden layers, an output layer, and an input layer where both linear and non-linear functions are applied in specific configurations. Within these hidden layers, feature maps are generated as a result of the outputs from hidden units. This is achieved by applying the output via preceding layer to a convolutional filter that has predefined weight and volume parameters. An activation function is applied to the result after multiplying the convolutional filter kernel. In terms of math, Equation 1 explains how to compute the ith layer's output, which is denoted by x^i.

$$x_j^i = f\left[\sum_{i=0}^{j} x_i^{j-1} * w_{ij} + b_j^i\right] \quad (1)$$

The variable x in the equation above represents the output of the final layer, often known as the feature map. The variable w represents the weight of the convolution kernel for the i^{th} layer, while the variable f indicates the activation function that is applied to the convolutional operation's output. Moreover, b indicates the bias specific to the i^{th} layer. Equation 2 offers a quantitative assessment that ascertains the result of the pooling layers and enhances the feature representation even further.

$$Q = 1 + \frac{2P + N - F}{S}$$

(2)

The height and width of the input image are represented in pixels by its dimension, which is denoted as N x N. In order to modify the final feature map's size, padding depth, or P, is a technique that adds more pixels to the surrounding area of the input image. The dimensions of the convolutional filter, represented as F x F, determine the height and width of these filters. As a result, significant traits are extracted from the input. Finally, the number of pixels that the filter is shifted both vertically and horizontally throughout the convolution process is determined by the stride parameter, which is represented by the letter S. Stride has a significant impact on the spatial dimensions of feature maps as well as the level of information obtained during the feature extraction procedure [20].

Values that are near together are given specific weights in an analysis process called the convolutional operation. The weighted values of these neighbors are then added together to determine the value of the current input, or pixel. When applied to a 2D input, such a picture, the process comprises recalculating the pixel values by summing the values of the surrounding pixels. This process uses a weight matrix, sometimes referred to as a kernel or filter, to compute these weighted sums. An activation function is a mathematical function that is applied to the output of a neuron or node in a neural network layer in the context of deep learning along with neural networks. It gives the network non-linearity, which enables it to recognize intricate patterns and produce predictions that get increasingly accurate.

CNNs use a method called pooling to abstract and generalize the features that the convolutional filters have collected, enabling the network to recognize pertinent features regardless of where they are located within a picture [21]. A 2-D filter is methodically moved across each the feature map's channel during the pooling procedure, and the features that fall inside the filter's region are combined or summarized. Upon implementing max-pooling, which preserves the highest value within the filter's area, the output's dimension (J) is ascertained as follows:

$$J = [1 + a_x - m] \times [1 + a_y - n] \times a_z$$

(3)

The output dimensions post-max-pooling for a feature map with dimensions of ax * ay * az will be smaller, usually as a result of down-sampling, while maintaining the most important features. The size of the pooling filter, the stride taken, and the particular pooling procedure utilized all affect how big the output will be. A common pooling technique in CNNs is called max pooling, and it tries to abstract and down-sample the data found in feature maps. After dividing the input data into those sections, it selects the maximum value inside each non-overlapping zone. The max pooling mathematical expression is as follows:

$$P[i, j] = \max\left(M\left[p*i : p*(i+1), q*j : q*(j+1)\right]\right)$$

(4)

Here, *P* [i, j] is the value at the i^{th} row and j^{th} column of the pooled feature map, and it represents the maximum value within the corresponding region defined by the pooling window. For input feature map (*M*) of size (*h* x *w*), a max pooling operation with a window (or filter) of size (*p* x *q*), and the outcome of the maximum pooling process (*P*) will have dimensions (*m* x *n*). Flattening is a critical operation in neural networks, frequently employed as a bridge between fully linked and convolutional layers. Within convolutional layers, the data is represented as feature maps, which are two-dimensional grids. However, fully connected layers require one-dimensional input. The flattening operation transforms these two-dimensional feature maps into a one-dimensional vector while preserving the order of the elements. This operation essentially "unstacks" the feature maps, concatenating all their values into a linear array. Finally *softmax* function is used to separate the features and obtain the classes.

B. Extreme Learning

This research leverages Extreme Learning (EL) as a tool to enhance classifier optimization. EL is attached within a CNN framework, offering the advantages of accelerated learning, smoother convergence, and reduced randomness. The result of the EL process is the classification of Plant Village dataset classes. After the initial EL phase's training is complete, the output from the first The buried layer is then transferred to the second EL after EL is discarded. Given that the hidden layer is configured with a fixed 2048 neurons, this EL instantiation represents a sparse interpretation. The EL's weights are fine-tuned utilizing the Adaptive Moment Estimation (Adam) optimization algorithm.

Assuming that the EL-CNN's hidden layer contains N_h neurons, the output vector from this layer, designated as $h(x_i)$ for a given input vector x_i, possesses a 1 x N_h size. The weight vector that connects the hidden layer to the output layer, represented by α, has dimensions of N_h x N_o. The number of output classes is shown here by the letter No. The output of EL is explicitly defined by equation 5.

$$f(x_i) = \alpha \times h(x_i), \quad i \in \{1 : N_h\}$$

(5)

The objective function, which serves the purpose of minimizing network errors, is explicitly defined in Equation 6.

$$f_{loss} = \Omega \sum_{i=1}^{N_h} \left\| E_i \right\|^2 + \min \left\| \alpha \right\|_g^2$$

(6)

In this context, the symbol ||α|| denotes the weight vector's Frobenius norm, and Ω the penalty constraint. During the training phase, the error vector is represented by the letter E. Figure 4 shows the EL-based learning model's step-by-step flow.

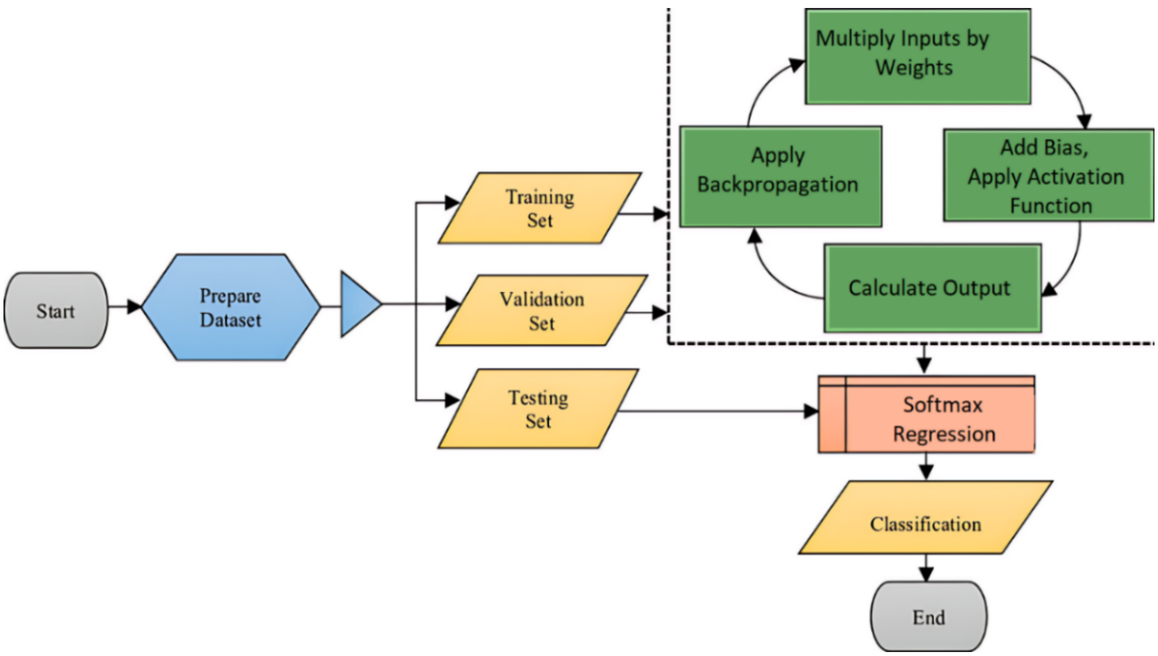


Figure 4: Proposed EL Scheme flow chart

Linear system's minimal average response is used in the EL theory to statistically figure out the output weights, whereas the input weights are generated randomly following a continuous distribution function [22]. This study presents a CNN classifier grounded in the principles of EL, aiming to elevate the performance beyond a basic EL model and unleash its full potential. The intricate approach outlined in Algorithm 1 encompasses *N* distinct CNNs, with the ultimate decision being derived from the amalgamation of diverse outcomes using a parameter set denoted as *F*, which is determined through an EL algorithm.

Algorithm 1: Proposed EL Classifier Training
Input: X=Dataset, T= Target, L= Hidden nodes, N= Models. **Output:** Parameters of EL Classifier.
Step 1: Initialize the value of **n=**1**, ….. ,** ***N*****.**

Step 2: Generate input weights through a random selection process $W^{(n)}$ and bias $b^{(n)}$.
Step 3: Compute the hidden matrix, $H = [b^{(n)} + W^{(n)} X]G$
Step 4: Compute the output weights, $\alpha^{(n)} = H^{(n)} \oplus T$
Step 5: Obtain the output, $Y^{(n)} = H^{(n)} \alpha^{(n)}$
Step 6: Find the global hidden matrix using Equation, $H_g = \left[Y^{(1)} Y^{(2)} .... Y^{(N)} \right]$
Step 7: Compute the fusion of parameters $F = H_g \oplus T$
Step 8: Return to the EL classifier parameters ($W^{(n)}$, $b^{(n)}$ and $\alpha^{(n)}$).
Step 9: Iterate through steps 2 to 8 repeatedly until the EL parameters reach convergence.

Table 2: Proposed EL-CNN Model Summary

Layers	Type	Output Shape	Parameters
Input Layer	Input	256 x 256 x 3	-
Convolution Layer	Conv2D	254 x 254 x 16	448
Max pooling layer	Maxpooling2D	127 x 127 x 16	0
Convolution Layer	Conv2D	125 x 125 x 32	4640
Max pooling layer	Maxpooling2D	62 x 62 x 32	0
Convolution Layer	Conv2D	60 x 60 x 64	18496
Max pooling layer	Maxpooling2D	30 x 30 x64	0
Dense	Dense	30 x 30 x 64	4160
Dense	Dense	30 x 30 x 32	2080
Flatten	Flatten	28800	0
Dense	Dense	6	172806
Total			202,630
Trainable			202,630
Non-Trainable			0

The EL-CNN model that has been suggested utilizes a layer layout where the input layer is the first layer. Convolutional, max pooling, dense, flatten, and completely linked layers are the levels that come after, layered on top of each other. To get improved classification outcomes, the model utilizes the Adam optimizer, which is effective for handling large datasets or parameter sets [23]. Hyperparameters in this model often have straightforward interpretations and are relatively easy to fine-tune. For the calculation of loss (error) Binary Cross Entropy (BCE) is used for both training and validation. With six classes, each calculated chance is compared using BCE to the real class output [24]. The probabilities are given a score according to how far they differ from the expected values. Table 2 lists the precise parameters that were used in this model to identify leaf diseases.

Proposed EL-CNN model designed for the Plant Village dataset is illustrated in figure 5, which begins with the initialization of 6 distinct classes. The model undergoes a series of processing steps, including convolutional layers, dense layers, max pooling layers, flattening, and hidden layers, ultimately resulting in a final output with six classes and predictions for individual classes. During the training and validation process, this model employed a batch size of 512 and a learning rate of 0.01, running for a total of 30 epochs. The choice of 30 epochs was made because it represents the optimal point at which the EL-CNN model converges, and the accuracy along with loss metrics stabilize, delivering the best results [25].

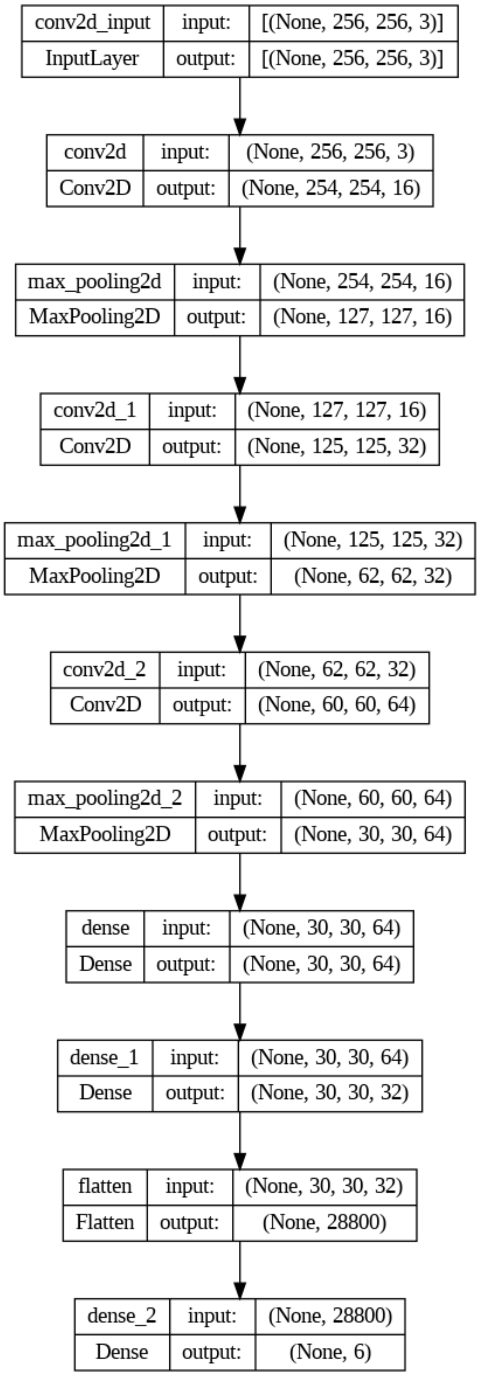
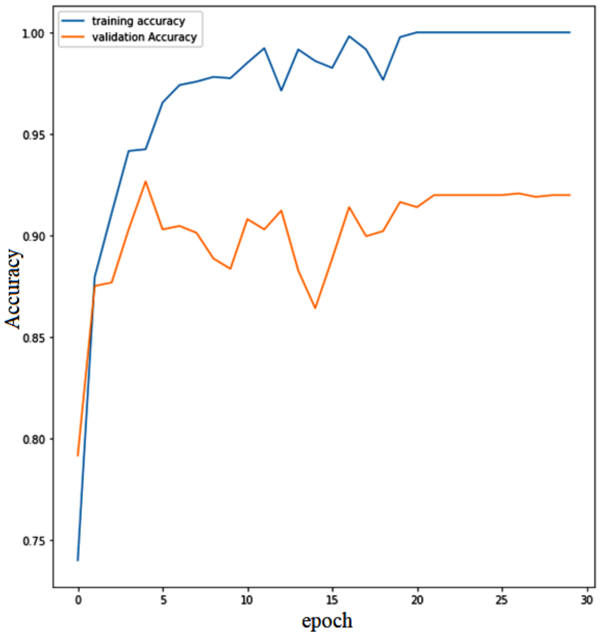


Figure 5: Proposed EL-CNN Model

4. Results and Discussions

The EL-CNN approach utilized in this research focuses exclusively on the deeper layers, keeping the layers before those layers fixed and training the classifier with the output features. This approach uses fewer entities for training, but it still requires a significant amount of features. For example, the feature set generated by the third convolutional layer (C3) has a size of 18,496 features. We used an 80:20 ratio in between testing data and training to set up this training technique. Furthermore, when comparing our results to previous research, we maintained consistency in our parameters, employing both cross-validation and fixed partitioning methodologies. Proposed EL-CNN model was implemented in Python and evaluated on the Google Colab platform. It's worth noting that a lower learning rate enhances the efficiency of EL-CNN training, while an excessively high learning rate can lead to training stagnation with unsatisfactory results. The suggested EL-CNN classifier's training and validation performance is shown in Figure 6.



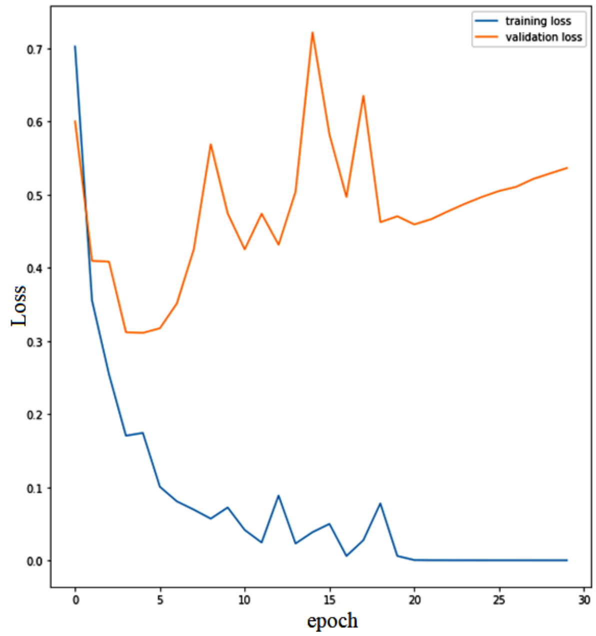


Figure 6: The proposed EL-CNN classifier's train and validation performance

In order to assess the validity and efficiency of the proposed EL-CNN model, we computed 4 key metrics: accuracy, recall, precision, and F1-score. These metrics are defined based on the usage of False Positive (P_f), False Negative (N_f), True Negative (N_t), and True Positive (P_t). The mathematical expressions for these performance parameters are as follows:

$$Precision = \frac{P_t}{P_f + P_t} \tag{7}$$

$$Recall = \frac{P_t}{N_f + P_t} \tag{8}$$

$$Accuracy = \frac{N_t + P_t}{N_f + N_t + P_f + P_t} \tag{9}$$

$$F1 - Score = \frac{2 \times P_t}{N_f + P_f + 2P_t} \tag{10}$$

From the 20th epoch onwards, the performance parameters consistently maintain high values. This stability is attributed to the effective application of EL technique to address the leaf disease categorization problem. Notably, the EL-CNN model's proposed loss, which is employed for the differentiation of leaf diseases, is impressively low, standing at 0.55. Moreover, the suggested EL-CNN model achieves an impressive 94.28% mean accuracy. Furthermore, the precision, recall, and F1-score mean values—which stand at 95.63%, 94.68%, and 96.23%, respectively—are also rather encouraging. The confusion matrix obtained for the multiclass classification model based on the suggested EL-CNN technique is shown graphically in Figure 7.

International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

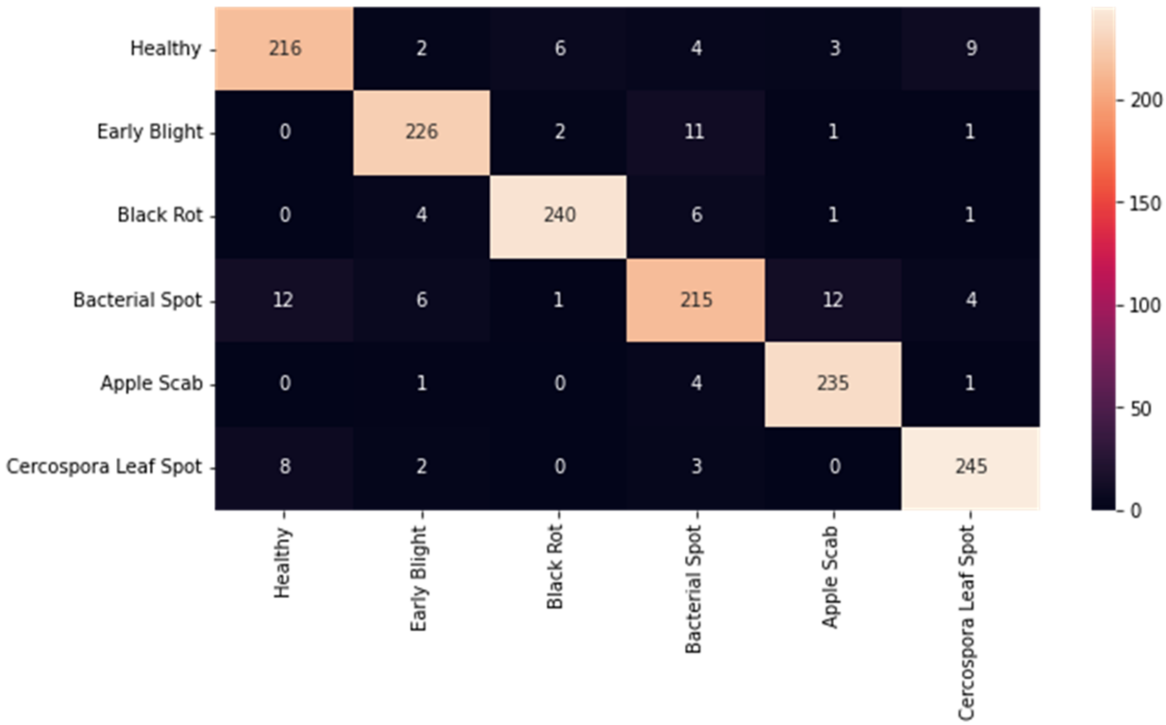


Figure 7: Confusion Matrix of Proposed EL-CNN Classifier

The suggested EL-CNN model performs well in identifying various leaf disease classes with better accuracy. It is clear from the classification report that EL-CNN performs well at differentiating between the various leaf disease classes. The Plant Village dataset can effectively identify the disease class with the use of the suggested EL-CNN model. Figure 8 shows the performance of each class individually.

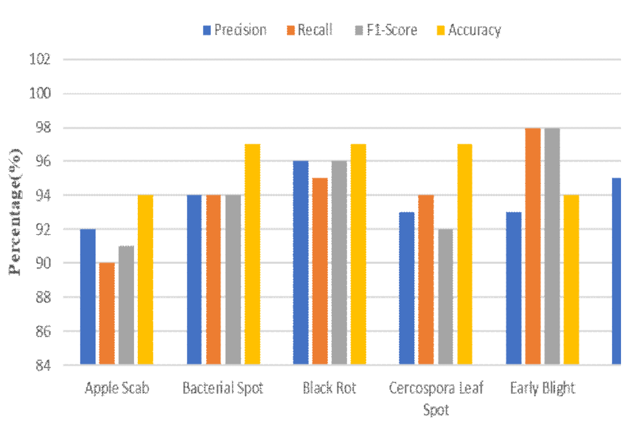


Figure 8: Performance of category-wise classification

A thorough analysis of the created EL-CNN method's classification performance is necessary to determine its efficacy. We conducted an evaluation of the classification performance of the various models on the Plant Village datasets. Based on the chosen performance criteria, we compare the effectiveness of the current models in Table 3.

Table 3: Comparison of Leaf Disease Classification Models.

Model	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
AlexNet	92.92	92.21	92.78	93.03
GoogleNet	90.73	91.09	89.56	90.31
ResNet 50	91.08	90.06	91.84	91.72
VGG16	91.27	91.42	90.37	91.33
Inception v3	90.92	89.06	89.74	90.81

E-mail:author's email

CNN	91.95	90.40	91.47	90.94
EL-CNN (Proposed)	95.63	94.68	96.23	94.28

While considering classification accuracy, the EL-CNN model leads with an impressive score of 94.28%. Among pre-trained models, we observe greater accuracy rates with VGG16 (91.33%), ResNet50 (91.72%), and AlexNet (93.03%). Notably, the accuracy of the proposed EL-CNN model surpasses that of AlexNet by 1.25%. Moving to precision, the EL-CNN model outperforms all other classifiers by offering a precision rate of 95.63%. In comparison, AlexNet achieves a precision of 92.92%, CNN attains 91.95%, and VGG16 scores 91.27%. The precision of EL-CNN exceeds that of AlexNet by 2.71%. With a recall value of 94.68%, the EL-CNN model has the highest value of all the models that were taken into consideration. In particular, VGG16 reports 91.42% recall, ResNet50 records 91.42%, and AlexNet records 92.78%. By contrast, the recall of the EL-CNN model is 3.45% higher than that of AlexNet. Moreover, the EL-CNN model is the best when evaluated based on the F1-score. AlexNet's F1-score is 93.03%, whereas the EL-CNN model achieves a remarkable 94.28%, indicating a 1.25% discrepancy between the two approaches. This emphasizes the significance of particular parameters and the part EL plays in lowering overfitting and raising classification accuracy. Notably, both AlexNet and the proposed model show good performance in detecting samples in big datasets. Figure 9 provides an overview of how EL-CNN compares to the most sophisticated leaf disease classifiers.

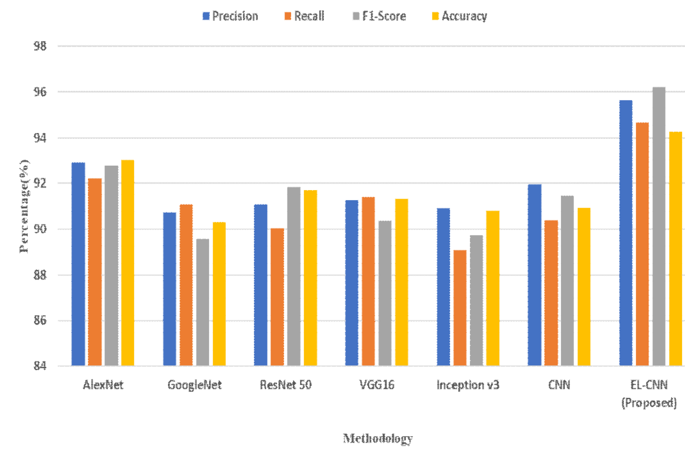


Figure 9: Comparison of Classification Performance

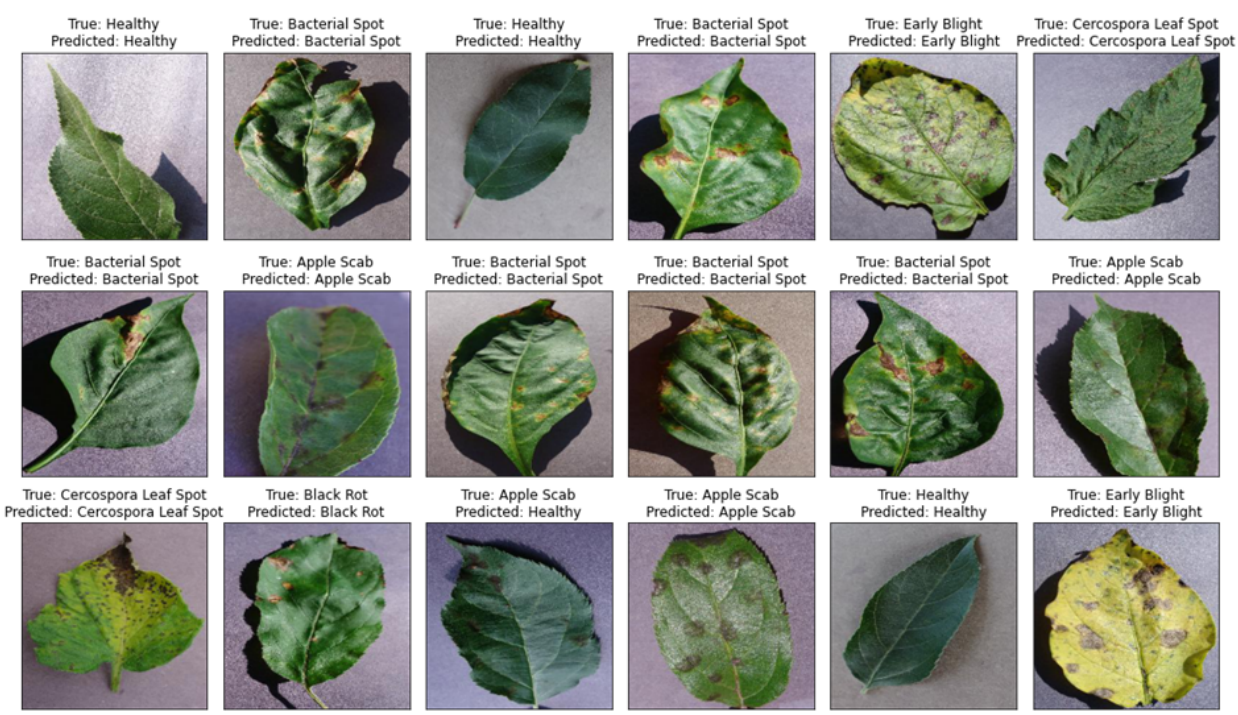


Figure 10: Ground truth and prediction results

ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

The efficiency of EL-CNN-based models lies in their utilization of extensive labeled datasets to tackle highly challenging problems. Employing EL-CNN systems in conjunction with large datasets for classification can automate the entire process of classification. Feature selection, ROI delineation, noise filtering, and feature extraction are no longer necessary. The EL-CNN models produce extremely reliable and bias-free predictions. Moreover, these models consistently achieve a remarkable level of accuracy, setting them apart from earlier CNN approaches. Leveraging GPU resources within the Google Colab framework as hardware significantly reduces the computation time. For instance, training the EL-CNN on the Plant Village dataset took just 4 minutes and 42 seconds. Crucially, the suggested multiclass classifier outperforms the current models in terms of performance metrics. A visual representation of sample predictions and the corresponding ground truth can be found in Figure 10.

5. Conclusion

This study looked examined EL-CNN in conjunction with a number of pre-trained CNN techniques for the image-based categorization of leaf diseases. Concatenating CNN structures with EL is an effective way to achieve the highest classification rate. With the Plant Village dataset, EL-CNN outperformed other classifiers, achieving 94.28% accuracy, 95.63% precision, 94.68% recall, and a 96.23% F1-score. The maximum prediction accuracy is produced by EL-CNN, thanks to optimization. It performs better than conventional strategies in removing the need for pre-processing phases. Additionally, the pre-trained AlexNet classifier produced worse performance metrics in comparison to the suggested EL-CNN. Future research will focus on integrating the models onto mobile platforms, minimizing computational complexity, and looking into different fine-tuning techniques.

Acknowledgement

The authors are thankful to concerned department of authors university for supporting the research.

References

1. T. Vijaykanth Reddy and K. Sashirekhak, "Examination on Advanced Machine Learning Techniques for Plant Leaf Disease Detection from Leaf Imagery," Journal of Critical Reviews, vol. 7, no. 5, pp. 1208-1221, 2020.
2. S. Kumbhar, A. Nilawar, S. Patil, B. Mahalakshmi, and M. Nipane, "Farmer buddy-web based cotton leaf disease detection using CNN," International Journal of Applied Engineering Research, vol. 14, no. 11, pp. 2662-2666, 2019.
3. M. Arsenovic, M. Karanovic, S. Sladojevic, A. Anderla, and D. Stefanovic, "Solving Current Limitations of Deep Learning Based Approaches for Plant Disease Detection," Symmetry, vol. 11, no. 7, 2019.
4. M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep Learning for Tomato Diseases: Classification and Symptoms Visualization," Applied Artificial Intelligence, vol. 31, no. 4, pp. 299-315, 2017.
5. S. Nema, B. Mishra, and M. Lambert, "Advance App Design Methods of Leaf Disease Detection Using Image Processing Approach," International Journal of Innovative Research in Science Engineering and Technology, vol. 6, no. 7, pp. 13277-13283, 2017.
6. K.P. Ferentinos, "Deep Learning Models for Plant Disease Detection and Diagnosis," Computers and Electronics in Agriculture, vol. 14, no. 5, pp. 311-318, 2018.
7. C. DeChant, N. Kaczmar, H. Lipson, M.A. Gore, and R.J. Nelson, "Millimeter-level plant disease detection from aerial photographs via deep learning and crowdsourced data," Frontiers in Plant Science, vol. 10, 1550, 2019.
8. S. Yadav, N. Sengar, A. Singh, A. Singh, and M.K. Dutta, "Identification of disease using deep learning and evaluation of bacteriosis in peach leaf," Ecological Informatics, vol. 61, 101247, 2021.
9. H. Durmuş, E.O. Güneş, and M. Kırcı, "Disease detection on the leaves of the tomato plants by using deep learning," in 2017 6th International conference on agro-geoinformatics, pp. 1-5, IEEE, 2017.
10. A. Voulodimos, N. Doulamis, G. Bebis, and T. Stathaki, "Recent developments in deep learning for engineering applications," Computational Intelligence and Neuroscience, vol. 7, pp. 31-43, 2018.
11. D. Tiwari, M. Ashish, N. Gangwar, A. Sharma, S. Patel, and S. Bhardwaj, "Potato Leaf Diseases Detection Using Deep Learning," in Proceedings of the International Conference on Intelligent Computing and Control Systems, ICICCS 2020, pp. 461-466, 2020.
12. R. Gandhi, S. Nimbalkar, N. Yelamanchili, and S. Ponkshe, "Plant disease detection using CNNs and GANs as an augmentative approach," in 2018 IEEE International

E-mail:author's email

Conference on Innovative Research and Development (ICIRD), pp. 1-5, IEEE, 2018.
13. R. Karthik, M. Hariharan, S. Anand, P. Mathikshara, A. Johnson, and R. Menaka, "Attention Embedded Residual CNN for Disease Detection in Tomato Leaves," Applied Soft Computing Journal, vol. 8, no. 6, pp. 105-117, 2020.
14. Y. Zhong and M. Zhao, "Research on Deep Learning in Apple Leaf Disease Recognition," Computers and Electronics in Agriculture, vol. 168, 105146, 2020.
15. C. Sullca, C. Molina, C. Rodriguez, and T. Fernandez, "Diseases Detection in Blueberry Leaves Using Computer Vision and Machine Learning Techniques," International Journal of Machine Learning and Computing, vol. 9, no. 5, pp. 656-661, 2019.
16. Ü. Atila, M. Uçar, K. Akyol, and E. Uçar, "Plant leaf disease classification using Efficient Net deep learning model," Ecological Informatics, vol. 61, pp. 101-121, 2021.
17. M.A. Azim, M.K. Islam, M.M. Rahman, and F. Jahan, "An effective feature extraction method for rice leaf disease classification," Telkomnika, vol. 19, no. 2, pp. 463-470, 2021.
18. H. Amin, A. Darwish, A.E. Hassanien, and M. Soliman, "End-to-end deep learning model for corn leaf disease classification," IEEE Access, vol. 10, pp. 31103-31115, 2022.
19. A. Bhujel, N.-E. Kim, E. Arulmozhi, J.K. Basak, and H.-T. Kim, "A lightweight Attention-based convolutional neural networks for tomato leaf disease classification," Agriculture, vol. 12, no. 2, pp. 228, 2022.
20. A.S. Paymode and V.B. Malode, "Transfer learning for multi-crop leaf disease image classification using convolutional neural network VGG," Artificial Intelligence in Agriculture, vol. 6, pp. 23-33, 2022.
21. N.K. Trivedi, V. Gautam, N. Goyal, and S. Kadry, "Early detection and classification of tomato leaf disease using high-performance deep neural network," Sensors, vol. 21, no. 23, 7987, 2021.
22. F. Al-Gaashani, F. Shang, M. Muthanna, M. Khayyat, and A.A. Ahmed, "Tomato leaf disease classification by exploiting transfer learning and feature concatenation," IET Image Processing, vol. 16, no. 3, pp. 913-925, 2022.
23. S.S. Chouhan, U.P. Singh, and S. Jain, "Automated plant leaf disease detection and classification using fuzzy based function network," Wireless Personal Communications, vol. 121, pp. 1757-1779, 2021.
24. U.K. Lilhore, A. Kumar, and C.-T. Li, "Enhanced convolutional neural network model for cassava leaf disease identification and classification," Mathematics, vol. 10, no. 4, pp. 580, 2022.
25. F.N. Al-Wesabi, M.A. Hamza, and A.S. Zamani, "Artificial Intelligence Enabled Apple Leaf Disease Classification for Precision Agriculture," Computers, Materials & Continua, vol. 70, no. 3, 2022.