INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

A Novel Deep Convolutional Neural Network Approach using Jacobi Polynomial and Laplacian Function (JPLF) in Recognition of Plant Leaf Disease

Pushparani S Janes[1], **P L Chithra**[2]*

1 Department of Computer Applications, Ethiraj College for Women, Chennai, Tamil Nadu, India
2 Department of Computer Science, University of Madras, Chennai, Tamil Nadu, India

OPEN ACCESS

Received: 24-10-2023

Accepted: 17-02-2024

Published: 03-04-2024

Citation: Janes PS, Chithra PL (2024) A Novel Deep Convolutional Neural Network Approach using Jacobi Polynomial and Laplacian Function (JPLF) in Recognition of Plant Leaf Disease . Indian Journal of Science and Technology 17(14): 1450-1463. https://doi.org/10.17485/IJST/v17i14.2651

* **Corresponding author**.

aswinchitra@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Background/Objectives: Enhancing agricultural productivity is crucial for fostering economic growth. Plant diseases significantly threaten crops, necessitating timely detection to mitigate adverse impacts on quality, quantity, and overall productivity. This research addresses the importance of early disease detection in agriculture and proposes an innovative method utilizing Jacobian Polynomial and Laplacian Function for precise identification. **Methods:** Efficient monitoring of large-scale crop farms with minimal workforce is essential. To achieve this, an automatic method for plant disease detection is proposed. The method leverages Jacobian polynomials to expand input features, mitigating correlation issues among input vectors. The expanded Jacobi polynomial is the input vector for a backpropagation algorithm with a novel activation function based on the Laplacian function. **Findings:** The efficacy of the proposed JPLF model is demonstrated through the accurate identification of leaf diseases, achieving a high testing accuracy of 92.07%. Comparative analysis with existing models, such as CNN with MobileNet V2 (85.38%) and the IoU model (83.75%), highlights the superiority of the JPLF model in plant disease detection. **Novelty:** To overcome the limitations of existing approaches, the incorporation of Jacobian polynomials plays a pivotal role in expanding input features. This expansion aids in eliminating correlations among input vectors, enhancing the efficacy of disease detection. The proposed model, Jacobi Polynomial and Laplacian Function (JPLF) introduces a unique activation function based on the Laplacian function, improving accuracy.

Keywords: Plant Disease Detection; Jacobi Polynomial; Laplacian Transform; Deep Learning Model; Feature Expansion

1 Introduction

In India, agriculture is a key source of economic growth. Agriculture-related industries constantly look for effective ways to safeguard crops from harm and improve food output. This drives researchers to look for novel, highly productive, and precise technologies. Crop diseases reduce output and financial losses for farmers and the agricultural sector. Disease identification is one of the key components of a successful farming system. In general, a farmer can spot disease symptoms in plants that require ongoing inspection by using eye observations. Various diseases can cause a plant's leaves to die. Farmers encounter extra challenges in diagnosing these illnesses. With photographs of plant leaves, image processing techniques are appropriate and effective for disease detection.

The traditional method of identifying plant illnesses involves direct eye observation and memory of the specific set of diseases according to the climate, season, etc. These techniques were time-consuming and imprecise. The most recent approaches for finding plant diseases need a variety of laboratory tests, knowledgeable personnel, well-equipped laboratories, etc. These items are not always accessible, especially in rural areas. Automatic disease detection is advantageous because it lessens the laborious task of keeping an eye on vast agricultural farms and identifies disease symptoms at an extremely early stage before they manifest on plant leaves. There are many methods to detect plant diseases[(1–3)]. If the disease has no external symptoms or appears only when it is too late to take measures, a complex examination is required.

The main technique for detecting plant diseases in practice is visual inspection by a trained specialist since most diseases show some form of visible manifestation. The variability of diseased plant symptoms can lead to misdiagnosis, as it can be more difficult to detect for experienced plant pathologists than for novice gardeners and hobbyists. Both hobby gardeners and professional experts would greatly benefit from an automated system that detects plant diseases based on plant appearance and visible symptoms as a validation system for disease diagnosis. The market for computer vision applications in precision agriculture is growing as a result of the development of computer vision[(4,5)]. These methods can be improved and extended.

Sharanya and Prashant[(6)] offer an effective way to determine if a tomato leaf is healthy or sick by extracting the leaf's texture using GLCM and extracting the colour of the histogram. A support vector machine (SVM) is trained using the gathered features in order to classify various plant diseases into communicable and non-communicable categories. After training, the model is installed on a Raspberry Pi, which uses a pi camera to record video and the classifying model to identify plant diseases in real time. The created system has an accuracy of 97.29% on the plant disease SVM model, which identifies two prevalent ailments, mold, and bacterial spot.

Regarding the identification and prediction of potato leaf diseases, Hritwik et al.[(7)] provide a thorough comparison and assessment of three cutting-edge CNN models: VGG19, DenseNet121, and ResNet50. It used a large dataset of pictures of potato leaves, with both healthy and diseased specimens, to train and evaluate the effectiveness of the selected CNN models. The variety and generalization capacities of the dataset were improved by the use of extensive data augmentation techniques. To identify the best model for practical applications, the models are assessed based on their accuracy, precision, recall, F1-score, and computing efficiency. The outcomes show that all three CNN models performed well in detecting and forecasting illnesses of the potato leaf, with VGG19 appearing as the best performer and DenseNet121 and ResNet50 following closely behind.

A new framework has been proposed by Chug et al.[(8)] that combines the advantages of both machine learning and deep learning. The framework contains 40 different hybrid deep learning models, which contain a combination of eight different pre-trained deep learning architecture variants, namely EfficientNet (B0–B7) as a feature extractor, and five machine learning techniques, viz. k-nearest neighbors. (kNN), AdaBoost, Random Forest (RF), Logistic Regression (LR) and Stochastic Gradient Boosting as classifiers. The Optuna framework was used in this study to optimize the hyperparameters of these classifiers. Real-time image data on tomato early blight was collected by the Indian Agricultural Research Institute for this study. The proposed HDL models perform exceptionally well on the IARI-TomEBD dataset and achieved high accuracy levels ranging from 87.55% to 100%. In addition, validation of the proposed approach was performed using two publicly available plant disease datasets, viz. PlantVillage-TomEBD and PlantVillage-BBLS. Finally, Friedman's statistical test was also performed to calculate the mean of the HDL models. The results show that EfNet-B3-ADB and EfNet-B3-SGB achieved the highest rank among all three plant disease datasets.

Another study presents an innovative deep learning technique for disease detection and classification called Ant Colony Optimization with Convolutional Neural Network (ACO-CNN) developed by Yousef et al.[(9)] The efficiency of plant leaf disease diagnosis was investigated using ant colony optimization (ACO). Color, texture, and geometry of plant leaf arrangement are extracted from the provided images using a CNN classifier. Some performance metrics used to analyze and propose the proposed method show that the proposed approach performs better than the existing techniques, and precise measures are used to implement these approaches. These steps are used in the stages of disease detection: image acquisition, image extraction, noise reduction, and classification.

Since DCNN has focused on agriculture fields in recent years, Gardie [10] applied an autonomous method in this research that employs CNN model to identify leaf diseases in tomato leaves. 18160 photos of tomato leaf diseases were used that were gathered from the Plant Village data collection. The dataset has been divided, with 40% designated for testing and 60% designated for training. By utilizing three color channels by applying various dropout values, augmentation, and segmentation approaches, the model obtained an accuracy of 96.8% for the dataset of 10 classes.

The advantages of the proposed JPLF method are as follows:

- This method automates non-contact observation of the health and development of herbs by using morphologic, textural, and time-based herbal features.
- The current work includes more diseases in plant species like cherries, corn, grapes, etc.
- This method uses Jacobian polynomials to expand the input feature in order to decrease the correlation between the input vectors.
- The expanded Jacobian polynomial serves as the input vector for the backpropagation algorithm, which makes use of a proposed activation function based on Laplacian polynomials.
- The application of Jacobian and Laplacian improves validation accuracy whereas
- Findings from conventional approaches are less accurate.

The rest of the paper is organized as follows: The dataset used for leaf disease classification and the proposed method is explained in section 2, while experimental setup and results are discussed in section 3. Finally, we draw our conclusion in section 4.

2 Materials and Methods

Deep learning is part of the algorithm of machine learning and artificial intelligence, where its layers are closely related to each other [11]. The result of the first layer is used as input to the next layer. In the plant disease detection experiment, a convolutional neural network is a suitable learning technique in deep learning, where it can accurately detect plant diseases [12]. This section describes the category of leaves that were the focus of this study as well as the methods used to get the images. Also, detailed here is the system configuration. An open-source resource called the plant village dataset [13] has 54,306 labeled images of healthy and sick leaves from 14 different plant species. Six distinct breeds of plants representing 18 different types of leaf diseases were used in this study's data collection. CNN prototypes were trained and tested using 17,103 images of both healthy and diseased plants. Dividing the total number of images arbitrarily so that 80% of them were considered as a training set and 20% as a test set, the complete database was divided into two datasets, a training set and a test set [14]. Table 1 summarizes the dataset, and some sample images of magazines used in this work are shown in Figure 1.

Table 1. Dataset Summary

Leaf	Disease	No. of Images
Cherry	Powdery mildew	1052
	Healthy	854
Corn	Cercospora leaf spot	513
	Common rust	1192
	Northern leaf blight	985
	Healthy	1162
Grape	Leaf blight	1076
	Black rot	1180
	Esca	1383
	Healthy	423
Peach	Bacterial spot	2297
	Healthy	360
Pepper Bell	Bacterial spot	997
	Healthy	1477
Potato	Early blight	1000
	Late blight	1000
	Healthy	152
	Total	**17,103**

In the realm of training machine learning and deep learning models, traditional algorithms have paved the way for various techniques aimed at optimizing model performance. Two commonly utilized loss functions include hinge loss and squared

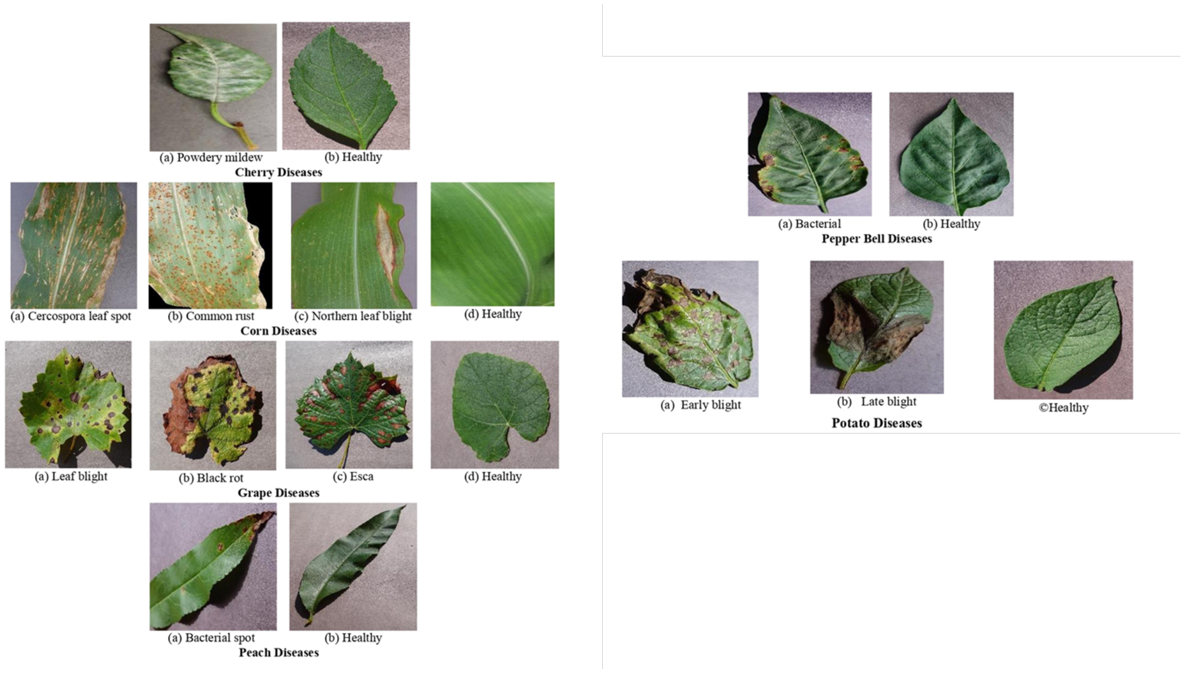

Fig 1. Healthy and Diseased Leaf Images

hinge loss, each providing distinct perspectives on measuring the efficacy of a model. Additionally, binary cross-entropy loss and softmax classifiers have gained prominence in the landscape of deep learning models, playing crucial roles in evaluating the classification accuracy of the dataset's data points.

Among the traditional algorithms, the backpropagation (BP) algorithm stands out as a foundational and widely adopted approach for implementing supervised learning tasks. This algorithm simulates a unidirectional mapping from an input space with r dimensions to an output space with s dimensions, where r and s represent the total number of input and output variables, respectively. At its core, the backpropagation algorithm leverages error feedback from training samples to iteratively update the weight links between layers, facilitating the learning process across the neural network. In the weight updation rule of the BP algorithm, the concept of the "learning rate" plays a crucial role. This hyperparameter influences the rate at which the network converges during training. The choice of an appropriate learning rate is pivotal; a too high value may risk overshooting the global minima, while an excessively low value can significantly slow down the network's learning pace.

To address the challenges posed by traditional algorithms, the proposed Jacobi Polynomial and Laplacian Function (JPLF) algorithm offers a novel solution. Traditional algorithms, such as those employing hinge loss, squared hinge loss, binary cross-entropy loss, and softmax classifiers, have their merits but may encounter issues related to convergence rates and the risk of getting trapped in local minima. The JPLF algorithm introduces a unique combination of Jacobi polynomials and Laplacian functions to mitigate these challenges (Figure 2).

2.1 Proposed Methodology

This JPLF method uses Jacobian polynomials to expand the input property to reduce the correlation between the input vectors. The extended Jacobian polynomial serves as the input vector to the back propagation algorithm, which uses a proposed activation function based on Laplacian polynomials.

2.1.1 Jacobi Polynomial

Mathematical analysis and real-world applications frequently make use of the classical Jacobi polynomials. The Legendre and Chebyshev polynomials in particular have been crucial to the development of spectral methods for partial differential equations[15].

Jacobian polynomials are a system of complete and orthogonal polynomials in mathematics that have a wide range of features that are helpful in numerous applications. Because of the problem's recurring relationship, Jacobian Polynomial based Neural

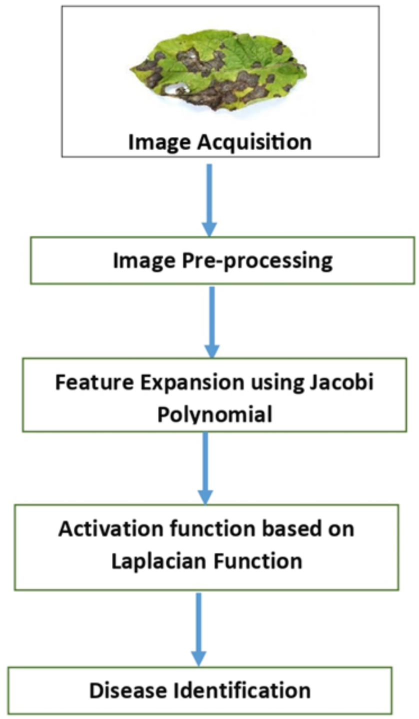

Fig 2. Block Diagram of Proposed Model JPLF

Networks were suggested in the field of artificial intelligence for function approximation. Compared to trigonometric equations, it requires fewer calculations. Figure 3 describes the structure of Jacobi polynomial.

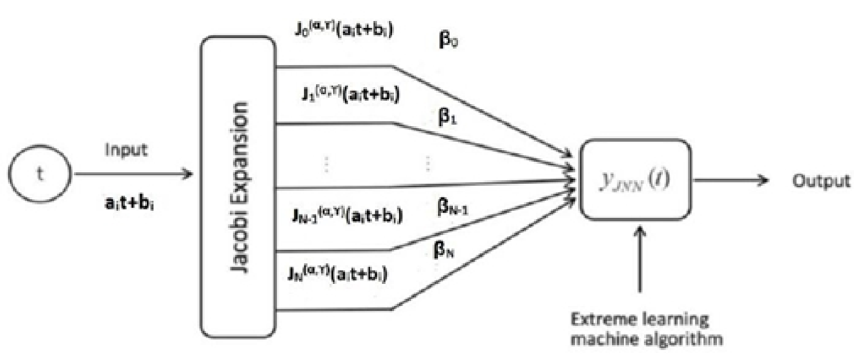

Fig 3. The Structure of Jacobi Neural Network

Definition: The definition of the Jacobi polynomials is introduced in this section.

Let $f(t)$, $g(t) \in \mathbb{C}[a,b]$, $\rho(t)$be the weight function in [a,b], if

$$(f,g) = \int_a^b \rho(t)\, f(t)\, g(t)\, dx = 0, \tag{1}$$

It is said that in [a, b], weighted functions f(t) and g (t) are orthogonal [16].

2.1.2 Laplacian Transform

This method is used to resolve differential equations. Here, an algebraic equation in the form of a frequency domain differential equation is first converted. To get at the differential equation's final solution, the algebraic problem must first be solved in the frequency domain and then the answer must be converted to time domain form. In other words, the Laplace transformation can be thought of as nothing more than a quick way to solve a differential equation.

For any t 0, let f(t) be the function of t. If this is done, the Laplace transform of f(t), F(s) can be expressed as

$$F(s) = \int_0^\infty f(t)\,.e^{-st} dt \tag{2}$$

Provided that the integral exists, where the Laplace Operator, s = σ + jω; will be real or complex j = $\sqrt{}$(-1).

The most well-known supervised machine learning algorithm is backpropagation (BP). The main issues with the BP algorithm have been noted to be poor convergence rates and local minima entrapment. Numerous scholars suggested various methods to address these issues. A novel algorithm (JPLF) is developed using both Jacobian polynomial and Laplacian functions. Jacobian polynomials are used in this JPLF approach to expand the input property and lower the correlation between the input vectors. The suggested activation function of the backpropagation method, which makes use of Laplacian polynomials, is based on the expanded Jacobian polynomial as the input vector. The steps are described in the following section.

Algorithm : Jacobian Polynomial and Laplacian Function (JPLF).

- **Initialization**

Step 1. Generate random weight matrices U and V between the input to the hidden layer and the hidden to the output layer, respectively

- **Jacobian Polynomial Expansion**

Step 2. For each training pattern set T = {X_i,O_i }

where X_i is the input vector and O_i is the output vector

Step 3. Expand the input vector using Jacobian Polynomial:

$X_i = (a+1)(a+b+2)\left(\frac{X_i-1}{2}\right)$,

where a and b are hyperparameters.

- **Feedforward Process**

Step 4. Feed the expanded input vector into the input layer.

Step 5. Calculate the net input h_g for each hidden node:

$h_g = \sum_{\{i=1\}}^{n} u_{\{ig\}} * x_i$

where j=1,2,…,g.

Step 6. Compute the output for each hidden node using the proposed Laplacian activation function:

$g(h_g) = \frac{h_j}{\left((h_i)^2+(\alpha)^2\right)}$

where g(h_g) is the Laplacian activation function.

Step 7. Apply the softmax activation function to calculate the output for nodes in the output layer.

- **Backpropagation and Weight Updation**

Step 8. Calculate the gradient of the output node:

$g'(Z_k) = g(Z) * (1 - g(Z))$,

where k is the number of units in the output node.

Step 9. Compute the error at the output node:

$\delta_k = (O_k - g(Z_k)) * g'(Z_k)$.
Step 10. Calculate the error at the hidden node:
$\delta_g = g'(Z_k) * \sum_{\{k=1\}}^{m} \delta_k * V_{\{gk\}}$.
Step 11. Update the weights in the output layer:
$V(t+1) = V(t) + \alpha * \delta_k * h_g + \beta * (V(t-1))$,
where α is the learning rate, and β is the momentum.
Step 12. Update the weights in the hidden layer with momentum:
$U(t+1) = U(t) + \alpha * \delta_g * x_i + \beta * (U(t-1))$.

- **Convergence Criteria**

Step 13. Repeat steps 2-12 until the mean square error is reduced to 1 * 10^{-3}.

The proposed Jacobian Polynomial and Laplacian Function (JPLF) algorithm begins with the initialization of random weight matrices, denoted as U and V, connecting the input to the hidden layer and the hidden to the output layer, respectively. For each training pattern set $T = \{X_i, O_i\}$, where X_i represents the input vector and O_i is the corresponding output vector, the input vector is expanded using Jacobi polynomials in the Jacobian Polynomial Expansion step. The expanded input X_i is computed as $(a+1)(a+b+2)((x_i-1)/2)$, with a and b as hyperparameters. The feedforward process involves propagating the expanded input through the network, calculating the net input h_g for each hidden node and determining the output for each hidden node using the Laplacian activation function $g(h_g) = h_j/(((h_i)^2 + (\alpha)^2))$. The softmax activation function is then applied to calculate the output for nodes in the output layer. Subsequently, backpropagation and weight updation steps involve computing gradients, errors, and updating weights in both the output and hidden layers using a combination of learning rate α and momentum β. The algorithm iteratively repeats these steps until the mean square error is reduced to $1 \times 10^{\{-3\}}$, serving as the convergence criterion. This comprehensive process integrates Jacobi polynomials and Laplacian functions to enhance the network's learning and classification capabilities. The architecture of the algorithm is given in Figure 4.

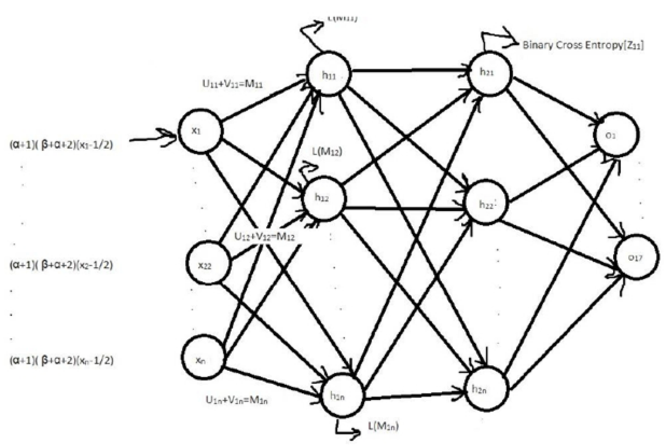

Fig 4. Proposed Architecture

3 Results and Discussion

The research employs Python and OpenCV due to their user-friendly nature and seamless integration, making them suitable for inclusion in OpenCV applications. The investigation utilizes a dataset consisting of 17,104 images from the Plant Village dataset. The proposed system's performance is evaluated based on accuracy.

The Table 1 presents a comprehensive overview of the distribution of leaf images across different plant species and their corresponding health conditions. The dataset encompasses four plant categories: Cherry, Corn, Grape, and Peach, each associated with specific diseases and a healthy state. For instance, in the Cherry category, images are categorized into Powdery Mildew and Healthy conditions. Similar categorizations are observed for Corn (Cercospora Leaf Spot, Common Rust, Northern Leaf Blight, and Healthy), Grape (Leaf Blight, Black Rot, Esca, and Healthy), and Peach (Bacterial Spot and Healthy).

Additionally, there are entries for Pepper Bell and Potato, featuring Bacterial Spot and Healthy, and Early Blight, Late Blight, and Healthy, respectively. The total number of images in the dataset amounts to 17,103, providing a diverse and extensive collection for plant disease detection and analysis.

3.1 Neural Network Architecture

The architecture adopts the VGG16Net design, featuring three fully interconnected levels: FC6, FC7, and FC8. FC6 returns 4096 features, FC7 returns 2622 features, and FC8, responsible for global characteristics, contains 1000 features. FC6 and FC7 focus on local characteristics, providing detailed pixel information, with FC7 contributing to reduced compilation complexity. The achieved maximum accuracy is documented in Table 2.

Table 2. Simulation Results for Proposed JPLF and BPN

Methodology	Level of Feature	Training Accuracy	Testing Accuracy
Raw Input with BPN	FC7 (Local)	80%	74%
Prop. Jacobian Expanded Input with BPN	**FC7 (Local)**	**85%**	**82%**
Raw Input with Laplacian Activation Function	FC7 (Local)	84%	82%
Prop. Jacobian Expanded input with Laplacian Activation Function	**FC7 (Local)**	**96.32%**	**92.07%**

3.2 Feature Transformation

The normal input raw data undergoes transformation using Jacobian polynomial, rendering features independent of each other. Additionally, Laplace transform, an integral transform converting real-variable functions into complex-variable functions, is applied. The activation function in hidden layers is based on the Laplacian sigmoid activation method (proposed), while the output layer employs softmax activation, given the multiclass nature of the problem.

3.3 Gradient Function and Error Handling

The activation function's expansion, treated as a Laplace transform, and the gradient function are collinearly aligned with 13 different output classes, functioning as an error function. This approach ensures effective handling of errors in the neural network.

The detailed integration of Python, OpenCV, VGG16Net, Jacobian polynomial, Laplace transform, and the proposed activation functions contributes to the system's overall accuracy and performance in plant disease detection.

Figure 5 shows the intermediate feature maps for the first convolutional layer of a VGG16 model trained on a plant disease detection dataset. The feature maps are arranged in a grid, with each square representing a single feature map. The feature maps for the first convolutional layer typically detect simple features in the input image, such as edges and corners. In Figure 5, we can see that some of the feature maps are more active than others. This suggests that the model has learned to detect certain features in the plant images that are more important for disease detection.

Figure 6 shows the intermediate feature maps for the second convolutional layer of a VGG16 model trained on a plant disease detection dataset. The feature maps are arranged in a grid, with each square representing a single feature map. The feature maps for the first two convolutional layers typically detect more complex features than the feature maps for the first convolutional layer, such as combinations of edges and corners. In Figure 6, we can see that some of the feature maps are more active than others. This suggests that the model has learned to detect certain features in the plant images that are even more important for disease detection than the features that are detected by the first convolutional layer.

Figure 7 shows the intermediate feature maps for the third convolutional layer of a VGG16 model trained on a plant disease detection dataset. The feature maps are arranged in a grid, with each square representing a single feature map. The feature maps for the third convolutional layer typically detect even more complex features than the feature maps for the first two convolutional layers, such as combinations of edges, corners, and textures.

Figure 8 shows the intermediate feature maps for the fourth convolutional layer of a VGG16 model trained on a plant disease detection dataset. The feature maps are arranged in a grid, with each square representing a single feature map. The feature maps for the fourth convolutional layer typically detect even more complex features than the feature maps for the first three convolutional layers, such as combinations of edges, corners, textures, and shapes.

Figure 9 shows the intermediate feature maps for the fifth and final convolutional layer of a VGG16 model trained on a plant disease detection dataset. The feature maps are arranged in a grid, with each square representing a single feature map. The

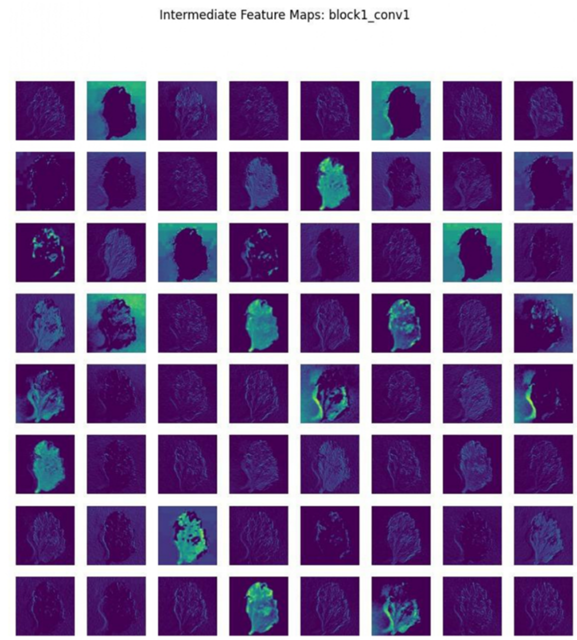

Fig 5. Intermediate Feature Maps Block1_Conv1

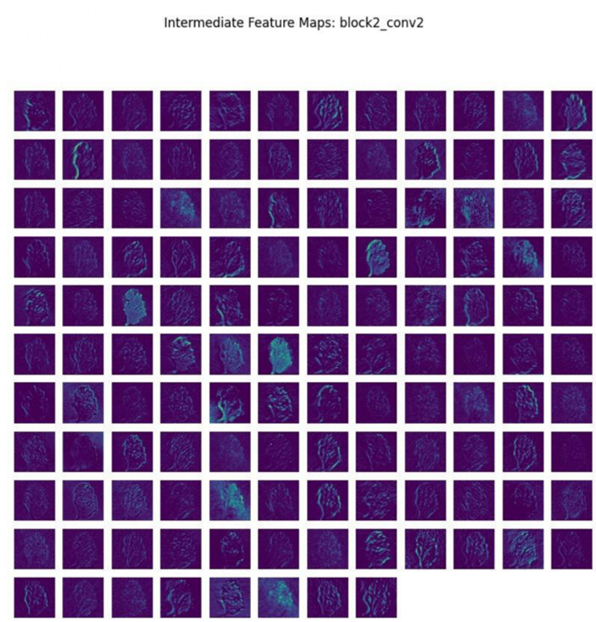

Fig 6. Intermediate Feature Maps Block2_Conv2

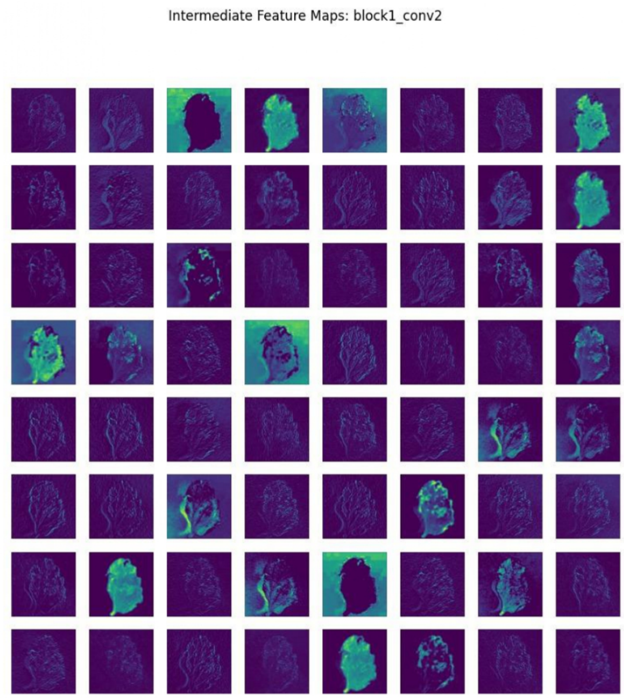

Fig 7. Intermediate Feature Maps Block1_Conv2

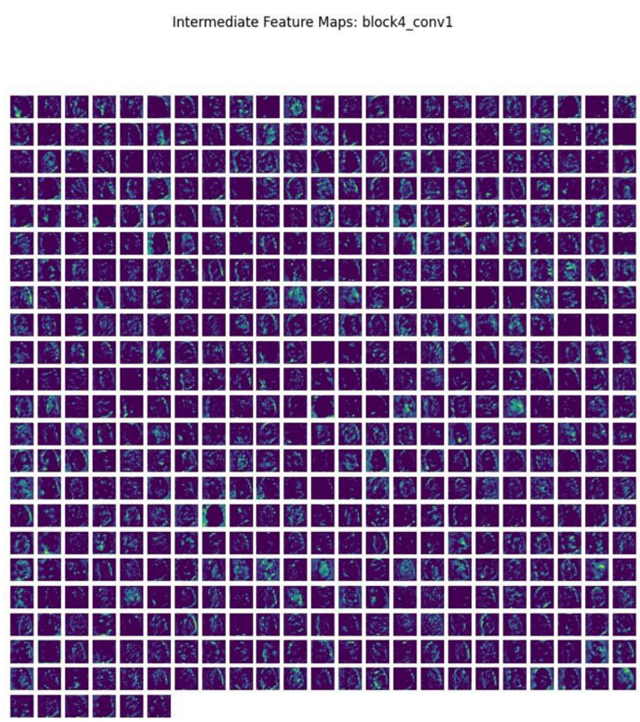

Fig 8. Intermediate Feature Maps Block4_Conv1

feature maps for the fifth convolutional layer typically detect the most complex and abstract features in the input image, such as the overall shape of the leaf, the distribution of leaf veins, and the presence of certain types of leaf lesions.

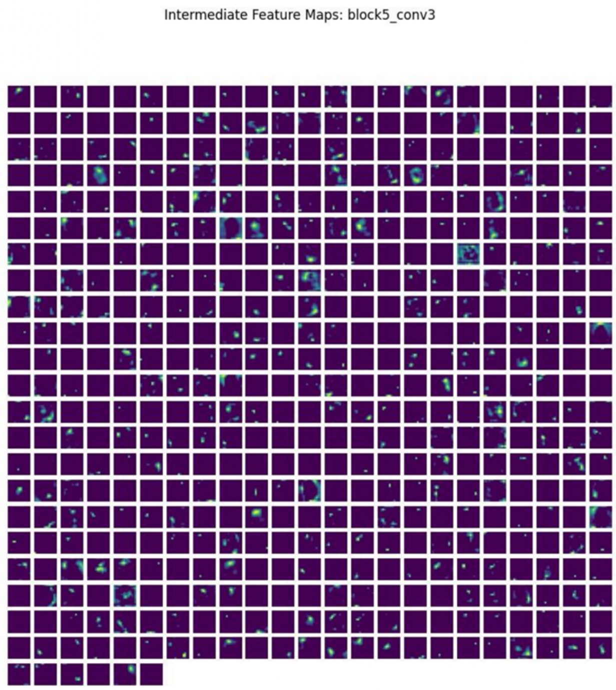

Fig 9. Intermediate Feature Maps Block5_Conv3

The analysis of the intermediate feature maps revealed that the VGG16 model is able to learn important features for plant disease detection. The model pays attention to changes in leaf color, texture, shape, vascularity, and lesion patterns, which are all important indicators of plant disease. This information can be used to develop new and improved plant disease detection systems.

3.4 Performance of the Proposed JPLF

The performance of the two algorithms, existing BPN to the proposed JPLF on a single dataset is represented in Figure 10. The comparison depends on the accuracy. Existing techniques and proposed method were applied to the singular dataset that contains enormous features for prediction. Based on accuracy, the proposed algorithm performed better than the existing techniques.

The output parameters utilized in this work is grounded on the complete categorization accuracy, the precision, the recall, and the F1-score. Accuracy is regarded as one of the significant measures exhibiting the output of a classifier. The realization ratio is calculated as the rate of the number of perfectly categorized attempts to the complete number of attempts for a true positive TP (a hit), a true negative TN (correct rejection), a false positive FP is said to be the end result which is falsely predictable to be certain, and a false negative FN is said to be the end result which is falsely predictable to be low however it occurs to be truly positive.[17]

3.4.1 Accuracy

The % of true expectations for the trial dataset is referred to as accuracy. This is easy to calculate by dividing the total number of forecasts by the number of correct guesses depicted in Equation (12).

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{12}$$

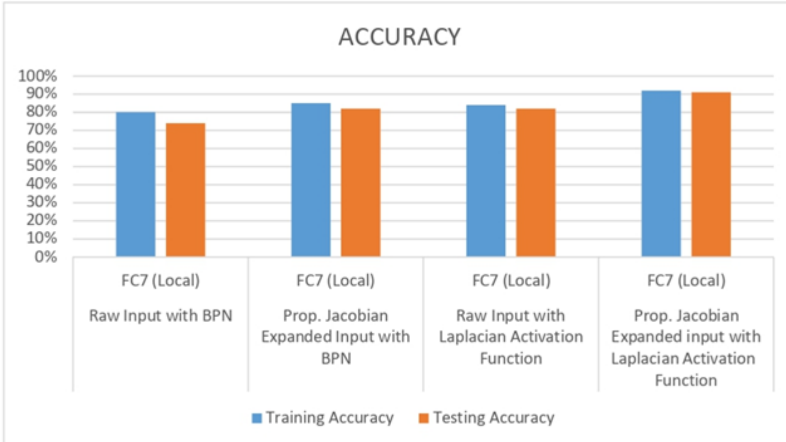

Fig 10. Comparative Chart of JPLF with BPN

3.4.2 Precision

Precision is referred to as the percentage of suitable models (true positives) amongst each and every model expected to be part of a specific group as shown in Equation (13).

$$Precision = \frac{TP}{TP+FP} \tag{13}$$

3.4.3 Recall

The division of models expected to be part of a group compared to each and every model which genuinely be part of a group is known as recall represented in Equation (14).

$$Recall = \frac{TP}{TP+FN} \tag{14}$$

3.4.4 F1 – Score

The F1 score is defined as the harmonic mean of precision and recall presented in Equation (15).

$$F1 = \frac{2 \times Precision \ \times Recall}{Precision + Recall} \tag{15}$$

The results in Table 2 reveal the effectiveness of different methodologies in the plant disease detection system, assessed based on training and testing accuracy. The conventional approach utilizing raw input with Backpropagation Neural Network (BPN) and FC7 (Local) features achieved a moderate training accuracy of 80% but showed a lower testing accuracy of 74%. Introducing the proposed Jacobian expanded input with BPN improved both training and testing accuracy to 85% and 82%, respectively. Similarly, utilizing the raw input with a Laplacian activation function for FC7 (Local) features demonstrated competitive training and testing accuracies of 84%. However, the most notable performance was observed with the proposed Jacobian expanded input coupled with the Laplacian activation function, achieving a remarkable training accuracy of 96.32% and impressive testing accuracy of 92.07%. These results underscore the efficacy of the proposed feature extraction methodology in significantly enhancing the accuracy of plant disease detection compared to traditional approaches.

It has become ever growing trend over the past few years to combine increasingly ”deep” networks with advanced Legendre Polynomial techniques to create solutions that perform steadily better. Single networks can now perform better by getting around some of their existing constraints by combining deep learning networks with more conventional methods[18]. The performance metrics in Table 3 demonstrate the efficacy of the proposed plant disease detection system, showcasing a remarkable accuracy of 96.32% during training and 92.07% during testing. The precision, recall, and F1-score further emphasize the system’s robustness in accurately identifying and classifying instances of plant diseases. During training, the model achieved a precision of 98.88%, indicating a low false-positive rate, and a recall of 94.83%, highlighting its ability to capture a high proportion of actual positive cases. The balanced F1-score of 96.81% underscores the model's overall reliability. In the testing phase, the system maintained a high precision of 94.57%, recall of 90.56%, and a balanced F1-score of 92.52%, confirming its effectiveness in providing accurate and well-rounded predictions on new, unseen data. These metrics collectively attest to the robust performance of the proposed methodology in plant disease detection.

Table 3. Performance Analysis of Proposed JPLF

	Accuracy	Precision	Recall	F1 - Score
Training	96.32	98.88	94.83	96.81
Testing	92.07	94.57	90.56	92.52

3.5 Performance Analysis with the existing work

Table 4 provides a comparative analysis of the accuracy of the proposed plant disease identification model with that of other models in the literature. The models include LeafNet by Chen et al.[19] comprising 7 classes provided an accuracy of 90.16%, whereas the CNN with MobileNet V2 by R. Surya et al.[20] with 5 classes gave an accuracy of 85.38% and ELM by Xian TS. et al.[21] yielded an accuracy of 84.94% incorporating 10 classes. The model IoU by Guo et al.[22] using 4 classes resulted with an accuracy of 83.75%, and K means and SVM by Bhange et al.[23] obtained an accuracy of 82% using 2 classes. The proposed model, JPLF, introduced in 2023, surpasses these models with an expanded class range of 17 and achieves a notable accuracy of 92.07%. This comparison positions the proposed model as a highly effective and advanced solution for leaf disease identification, outperforming several contemporary models in the field.

Table 4. Comparison of the accuracy of the proposed model with that of the latest published Leaf Disease Identification model

Name	Year	No, of classes	Model	Accuracy in %
Chen J, et al.[19]	2019	7	LeafNet	90.16
R. Surya. et al.[20]	2020	5	CNN with MobileNet V2	85.38
Xian TS. et al.[21]	2021	10	ELM	84.94
Guo, et al.[22]	2020	4	IoU	83.75
Bhange et al.[23]	2015	2	K means and SVM	82
Proposed Model	**2023**	**17**	**JPLF**	**92.07**

4 Conclusion

The paper highlights the growing academic interest in deep learning techniques due to their powerful learning capabilities and effectiveness in handling intricate patterns[24]. In response to the challenges of classification, detection, and segmentation problems, the study introduces a method for leaf disease detection using a convolutional neural network (CNN) with Jacobian Polynomial and Laplacian Function (JPLF). The approach involves utilizing a pre-trained CNN model to expand input vector characteristics into Jacobian polynomials, subsequently employing a modified Backpropagation Algorithm with a Laplacian-based activation function.

Comparative analyses against existing models such as LeafNet, CNN with MobileNet V2, IoU, and K Means with SVM reveal the superior performance of JPLF in recognizing leaf diseases. The paper suggests that JPLF holds promise for future applications, offering improved efficiency and accuracy in plant disease diagnoses. The reported high accuracies of the proposed method compared to other models affirm its capability in accurately identifying and categorizing various plant diseases within the dataset. Additionally, the paper acknowledges the potential for further enhancement by increasing the epoch value, albeit at the expense of processing time, and suggests expanding the dataset to accommodate a broader range of plants and diseases, thus enhancing the adaptability of the plant disease identification procedure.

References

1) Esgario JGM, Krohling RA, Ventura JA. Deep learning for classification and severity estimation of coffee leaf biotic stress. *Computers and Electronics in Agriculture*. 2020;169:105162. Available from: https://doi.org/10.1016/j.compag.2019.105162.

2) Ozguven MM, Adem K. Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms. *Physica A: Statistical Mechanics and its Applications*. 2019;535:122537. Available from: https://doi.org/10.1016/j.physa.2019.122537.
3) Ma J, Du K, Zheng F, Zhang L, Gong Z, Sun Z. A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Computers and Electronics in Agriculture*. 2018;154:18–24. Available from: https://doi.org/10.1016/j.compag.2018.08.048.
4) Gensheng H, Haoyu W, Yan Z, Mingzhu W. A low shot learning method for tea leaf's disease identification. *Computers and Electronics in Agriculture*. 2019;163:104852. Available from: https://doi.org/10.1016/j.compag.2019.104852.
5) Coulibaly S, Kamsu-Foguem B, Kamissoko D, Traore D. Deep neural networks with transfer learning in millet crop images. *Computers in Industry*. 2019;108:115–120. Available from: https://doi.org/10.1016/j.compind.2019.02.003.
6) Kumar SP, and PKS. Image Processing And Machine Learning Approach For Tomato Leaf Disease Detection. *Journal of Survey in Fisheries Sciences*. 2023;10(4S):2641–2644. Available from: https://doi.org/10.17762/sfs.v10i4S.1623.
7) Ghosh H, Rahat IS, Shaik K, Khasim S, Yesubabu M. Potato Leaf Disease Recognition and Prediction using Convolutional Neural Networks. *EAI Endorsed Transactions on Scalable Information Systems*. 2023;10(6):1–8. Available from: https://doi.org/10.4108/eetsis.3937.
8) Chug A, Bhatia A, Singh AP, Singh D. A novel framework for image-based plant disease detection using hybrid deep learning approach. *Soft Computing*. 2023;27(18):13613–13638. Available from: https://doi.org/10.1007/s00500-022-07177-7.
9) Methkal Y, Algani A, Caro OJM, Bravo LMR, Kaur C, Ansari MSA, et al. Leaf disease identification and classification using optimized deep learning. *Measurement: Sensors*. 2023;25:1–6. Available from: https://doi.org/10.1016/j.measen.2022.100643.
10) Gardie B, Azezew K, Asemie S. Image-based Tomato Disease Identification Using Convolutional Neural Network. *Indian Journal of Science and Technology*. 2021;14(42):3126–3132. Available from: https://doi.org/10.17485/IJST/v14i42.1164.
11) What is deep learning and how does it work. 2021. Available from: https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neuralnetwork.
12) Tiwari D, Ashish M, Gangwar NM, Sharma A, Patel S, Bhardwaj S. Potato Leaf Diseases Detection Using Deep Learning. In: 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE. 2020;p. 461–466. Available from: https://doi.org/10.1109/ICICCS48265.2020.9121067.
13) Hughes DP, Salathe M. An open access repository of images on plant health to enable the development of mobile disease diagnostics. 2015. Available from: https://doi.org/10.48550/arXiv.1511.08060.
14) Kaya A, Keceli AS, Catal C, Yalic HY, Temucin H, Tekinerdogan B. Analysis of transfer learning for deep neural network based plant classification models. *Computers and Electronics in Agriculture*. 2019;158:20–29. Available from: https://doi.org/10.1016/j.compag.2019.01.041.
15) Fine TL. Feedforward Neural Network Methodology. 1st ed. Information Science and Statistics;New York, NY, USA. Springer. 1999. Available from: https://doi.org/10.1007/b97705.
16) Chihara TS. An introduction to orthogonal polynomials. 1st ed. New York, USA. Gordon and BreachScience Publishers, Inc. 1978. Available from: https://bayanbox.ir/view/1984196138202468281/Theodore.S.Chihara-An-introduction-to-orthogonal-polynomials.pdf.
17) Witten IH, Frank E, Hall MA, Pal CJ. Practical Machine Learning Tools and Techniques. In: Data Mining;vol. 2. Morgan Kaufmann. 2016. Available from: https://doi.org/10.1016/c2009-0-19715-5.
18) Ma M, Gao Z, Wu J, Chen Y, Zheng X. A Smile Detection Method Based on Improved LeNet-5 and Support Vector Machine. In: 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE. 2018;p. 446–451. Available from: https://doi.org/10.1109/SmartWorld.2018.00104.
19) Chen J, Liu Q, Gao L. Visual Tea Leaf Disease Recognition Using a Convolutional Neural Network Model. *Symmetry*. 2019;11(3):1–13. Available from: https://doi.org/10.3390/sym11030343.
20) Surya R, Gautama E. Cassava Leaf Disease Detection Using Convolutional Neural Networks. In: 2020 6th International Conference on Science in Information Technology (ICSITech);vol. 21. IEEE. 2021;p. 97–102. Available from: https://doi.org/10.1109/ICSITech49800.2020.9392051.
21) Xian TS, Ngadiran R. Plant Diseases Classification using Machine Learning. In: The 1st International Conference on Engineering and Technology (ICoEngTech) 2021;vol. 1962 of Journal of Physics: Conference Series. IOP Publishing. 2021;p. 1–12. Available from: https://iopscience.iop.org/article/10.1088/1742-6596/1962/1/012024/pdf.
22) Guo Y, Zhang J, Yin C, Hu X, Zou Y, Xue Z, et al. Plant Disease Identification Based on Deep Learning Algorithm in Smart Farming. *Discrete Dynamics in Nature and Society*. 2020;2020:1–11. Available from: https://doi.org/10.1155/2020/2479172.
23) Bhange M, Hingoliwala HA. Smart Farming: Pomegranate Disease Detection Using Image Processing. *Procedia Computer Science*. 2015;58:280–288. Available from: https://doi.org/10.1016/j.procs.2015.08.022.
24) Lee SH, Chan CS, Mayo SJ, Remagnino P. How deep learning extracts and learns leaf features for plant classification. *Pattern Recognition*. 2017;71:1–13. Available from: https://doi.org/10.1016/j.patcog.2017.05.015.