

Research Article

Hybrid Feature-Based Disease Detection in Plant Leaf Using Convolutional Neural Network, Bayesian Optimized SVM, and Random Forest Classifier

Ashutosh Kumar Singh ¹, SVN Sreenivasu ², U.S.B. K. Mahalaxmi ³,
Himanshu Sharma ⁴, Dinesh D. Patil ⁵ and Evans Asenso ⁶

¹Department of Electronics and Communication, SRK University, Bhopal, India

²Department of Computer Science and Engineering, Narasaraopeta Engineering College, Narasaraopet 522601, Andhra Pradesh, India

³Department of ECE, Aditya College of Engineering, Surampalem, India

⁴Department of Electronics and Communication Engineering, J B Institute of Engineering and Technology, Hyderabad, India

⁵Department of Computer Science and Engineering, Shri Sant Gadge Baba College of Engineering and Technology, Bhusawal 425203, India

⁶Department of Agricultural Engineering, School of Engineering Sciences, University of Ghana, Accra, Ghana

Correspondence should be addressed to Evans Asenso; [easenso@ug.edu.gh](mailto: easenso@ug.edu.gh)

Received 21 December 2021; Revised 31 December 2021; Accepted 3 January 2022; Published 10 February 2022

Academic Editor: Rijwan Khan

Copyright © 2022 Ashutosh Kumar Singh et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Plant diseases are unfavourable factors that cause a significant decrease in the quality and quantity of crops. Experienced biologists or farmers often observe plants with the naked eye for disease, but this method is often imprecise and can take a long time. In this study, we use artificial intelligence and computer vision techniques to achieve the goal of designing and developing an intelligent classification mechanism for leaf diseases. This paper follows two methodologies and their simulation outcomes are compared for performance evaluation. In the first part, data augmentation is performed on the PlantVillage data set images (for apple, corn, potato, tomato, and rice plants), and their deep features are extracted using convolutional neural network (CNN). These features are classified by a Bayesian optimized support vector machine classifier and the results attained in terms of precision, sensitivity, f-score, and accuracy. The above-said methodologies will enable farmers all over the world to take early action to prevent their crops from becoming irreversibly damaged, thereby saving the world and themselves from a potential economic crisis. The second part of the methodology starts with the preprocessing of data set images, and their texture and color features are extracted by histogram of oriented gradient (HoG), GLCM, and color moments. Here, the three types of features, that is, color, texture, and deep features, are combined to form hybrid features. The binary particle swarm optimization is applied for the selection of these hybrid features followed by the classification with random forest classifier to get the simulation results. Binary particle swarm optimization plays a crucial role in hybrid feature selection; the purpose of this Algorithm is to obtain the suitable output with the least features. The comparative analysis of both techniques is presented with the use of the above-mentioned evaluation parameters.

1. Introduction

Diseases, pests, and other undesirable substances present in crops can cause a sharp decline in agricultural production [1]. The impact of these dangerous factors on crops has a

direct impact on the decline of the quality and quantity of crops. To combat, control, and mitigate the effects of biological organisms and diseases, the term “pesticides” was coined [2]. Typically, the diagnosis of plant pests and diseases is usually analyzed by visual inspection based on the

appearance, morphology, and other characteristics of the leaves. It is suggested that this visual examination be performed and analyzed only by a highly trained biologist, as misdiagnosis can lead to irreparable loss of yield. It should be noted that pest and disease control research is usually costly and requires the presence of a specialized biologist to diagnose and prevent the spread and transmission of any disease as early as possible [3].

Recently, AI has found a large number of applications in day-to-day life, leading to the introduction of the terms “machine learning” (ML) and “deep learning” (DL), which, in terms of simplicity, allows machines to “learn” a large number of patterns and then take action. Machine learning and deep learning allow a software application to improve their prediction accuracy without being expressly designed to do so.

The link between DL technology and computer vision has led to the emergence of intelligent algorithms that analyze and classify patterns or images with more accuracy than the average person. DL is about machines learning to think using architecture modelled after the nervous system while computer vision is about computers learning to think and behave with minimal human intervention [4].

As a result of the above, computer technology has emerged for the automatic detection of plant diseases [5], which has developed a system for diagnosing foliage diseases using a smartphone using two network structures, AlexNet [6] and GoogLeNet [7]. The proposed model was trained and achieved 99.35% accuracy in diagnosing leaf diseases. For the analytical algorithm [8], they used the R-CNN diagnostic algorithm, which is faster compared to CNNs, to detect early-grown corns and distinguish them from weeds in three different climates with an average analytical result of 97.71%. In another review by Yu and others [9], using deep learning using the fastest sensor in conjunction with a ResNet50 neural network to detect fruits that would later be implemented in a strawberry-picking robot, the analysis results obtained in this study were with an accuracy of 95.78% and overlapped 89.85%. Fuentes and others [10] compared the three types of sensor families with respect to different architectures and achieved better results with the faster R-CNN together with the VGG-16 architecture.

“Plant diseases can cause great damage to agricultural crops by significantly reducing their production [11] because they restrict the growth of crops and lead to poor quality of products [12]. Due to its lower yield and fiber, biofuel crops as agriculture struggles to keep up with the world’s rapidly increasing population. An existing method for the detection and identification of plant leaf diseases is observed with the naked eye [13].” However, this manual recognition can have consequences as it can be misdiagnosed since the symptoms are judged according to their experiences [12]. Additionally, plants must be monitored in a consistent manner to avoid the spread of disease. This continuous monitoring represents a difficult task, considering that it requires a great amount of time [14]. Due to the above, computational techniques for automatic detection have emerged such as presented in [5], who developed a system for the diagnosis of plant leaf diseases assisted by smartphones. “They used a public data

set of 54,306 images of diseased and healthy leaves, employing two CNN architectures, AlexNet [6], and GoogLeNet [7]. Their proposed model was trained and reached an accuracy of 99.35% in the detection of leaf diseases. However, it performed poorly when tested on sets of images taken under different conditions.” On the other hand, Sladojevic and others [15] made a system based on CNN to identify 13 types of common diseases. Likewise, the results achieved a precision between 91% and 98%, obtaining a general average of 96.3%. Brahimi and others [14] applied a CNN model to classify tomato diseases based on leaf images. In order to analyze the deep model, they have used visualization methods to understand the symptoms and thus locate the regions of the leaf disease. The results obtained reached 99.18% accuracy. “In another research, Lu and others [16] proposed the use of CNN for the identification of 10 common rice diseases, using natural images of healthy and diseased rice leaves and stems captured from the experimental field. Finally, their model achieved an accuracy of 95.48%. Kawasaki and others [17] proposed the use of CNN to distinguish healthy cucumbers from infected ones through the use of leaf images. The system achieved an average accuracy of 94.9% in classifying cucumbers into two typical disease classes and one healthy class.” However, the aforementioned investigations apply a limited number of architectures. The present work proposes to evaluate the performance of pretrained CNNs: AlexNet, GoogLeNet, and InceptionV3 [18]; SqueezeNet [19]; ResNet50; and ResNet101 [20] in order to determine which classifies better and obtains results in less time. This study, by covering 5 types of crop species and 19 diseases, becomes a complete tool for those researchers who need to design and implement a mechanism that is sufficiently complete in the classification of diseases of plant leaves used in agriculture.

In this research paper, we compare the most common imaging algorithms for analyzing and classifying objects. Our work tries to simulate which algorithm predicts the best outcome when diagnosing the disease in plant leaves. It is expected that the results will be used to determine which algorithm is most effective in creating a smart system for detecting leaf diseases.

2. Proposed Methodology

This research paper presents the detection of leaf diseases in corn, apple, tomato, rice, and potato leaves by extraction of deep features and texture and color features, followed by feature selection based on BPSO, comparing the two classifications: Bayesian optimized SVM and random forest. Figure 1 provides a general flow chart of the proposed work.

3. Image Acquisition

A lot of information is required to train intelligent visualization and classification systems. In general, machine learning and deep learning systems improve their performance when training with large amounts of data. In this article, we have used the implementation of the PlantVillage database [21,22]. This “data set” consists of 54,323 images of

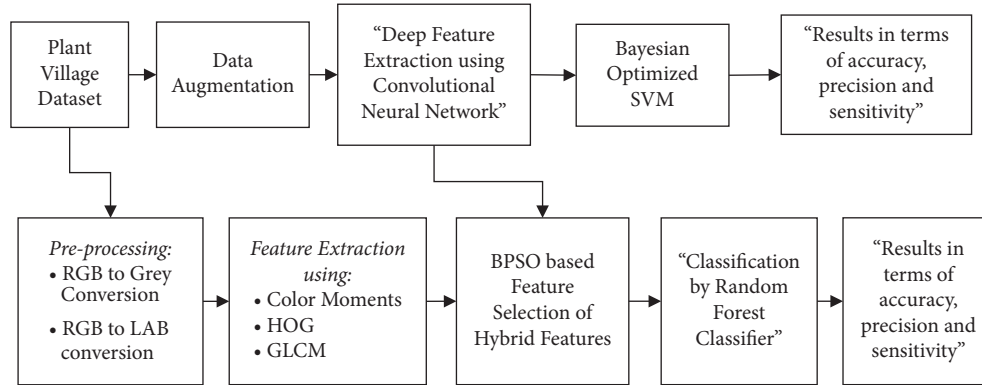


FIGURE 1: Block diagram for the hybrid approach of plant leaf disease detection.

14 plants and is divided into 38 groups of healthy leaves of plants with different types of diseases. This study used 37,315 images of 5 types of plants, apple (7,771 images), corn (7,316 images), potato (3,763 images), tomato (18,345 images), and rice (120 images).

To train and evaluate deep learning systems, the data volume should be divided into research sets and assessment sets. The data set generated from this study is unique in that it contains images of different sizes and provides more sensor power.

Plant leaf images were obtained from the PlantVillage data set [22] for preprocessing, feature extraction, feature selection, and classification. Tables 1–5 show examples of various leaf diseases of apple, corn, potato, tomato, and rice plants, respectively, from the data set.

3.1. Data Augmentation. This means that training the model must have enabled it to learn the main features of a data set. For this, the following are necessary:

- (i) Data Space: the learning data space covers the spectrum of possibilities, that is, it contains the largest number of different examples corresponding to the context of the use of the model
- (ii) Features Space: the feature space of the training data also covers the spectrum of possibilities, that is, it contains the greatest possible number of representations of each feature of the data

To satisfy the first point, we must therefore collect the greatest possible variety of training images corresponding to the context of use and the objective of our model.

And to satisfy the second point, we must apply data augmentation techniques to the training images at our disposal, the most common of which are affine transformations (horizontal and/or vertical flip, rotation). There are also nonaffine transformations such as, for example, variation in brightness and contrast, wrap (perspective), resizing, random crop (random part of an image), jitter (random noise), or cutout (squares random blacks).

3.2. Deep Features Extraction Using Convolutional Neural Network (CNN). CNNs after the initials are a special type of

neural network recommended for processing data with a network or grid topology. Images (a network of x and y pixels) are the most common types of data used in this type of network, but time series (data in 1-D with an additional dimension such as a time measurement) and three-dimensional data such as a scanner (two measurements associated with a 1-D image, most associated with the evolution of video in time) are also used [23].

CNN has been used with great success for many purposes. Recently, human vision has been replaced by image recognition through the use of deep convolutional neural networks [24]. The typical deep convolution neural network architecture is shown in Figure 2.

3.2.1. History and Progress. CNN has played a very important role in the history and development of artificial neural networks. This is a great example of using biological and physiological brain research (like CNN human vision) to develop artificial algorithms for machine learning and especially deep learning. They were also one of the first neural network models to achieve good results and performance and were used to develop commercial applications at the turn of the century. For example, in 1990 the AT&T research group developed an application to read accounts using a fixed neural network. In the late 1990s, this system read about 10% of US banknotes [25].

Over the years, Microsoft has developed a handwriting recognition model. One of the most recent and important advances in the application of convolutional neural networks occurred in 2012 when Krizhevsky won the ImageNet image collection competition [6], in which several images were divided into over a thousand different classes.

3.2.2. Convolution Operation. “The convolution is an operation applied to two functions with real numbers as arguments. The convolution operation is defined by the following mathematical expression:”

$$s(t) = \int x(a)w(t-a)da. \quad (1)$$

The convolution operation is usually denoted by:

TABLE 1: Apple plant leaf with disease (source: PlantVillage data set [22]).





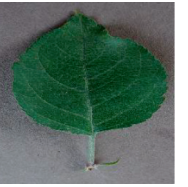










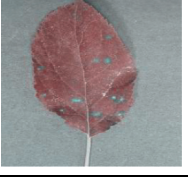


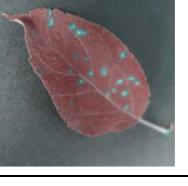
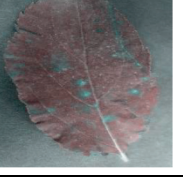
Healthy apple leaf					
Apple leaf with scab disease					
Apple leaf with black rot disease					
Apple leaf with cedar apple rust disease					

TABLE 2: Corn plant leaf with disease (source: PlantVillage data set [22]).

















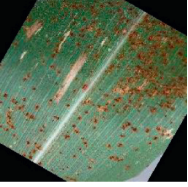








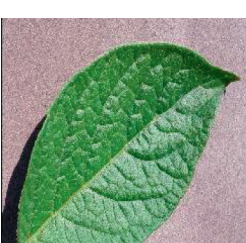
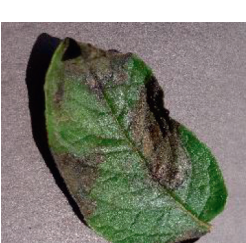
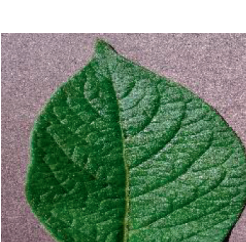
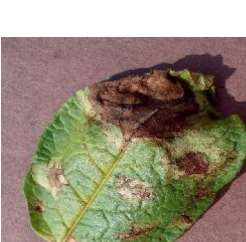
Healthy corn leaf					
Corn leaf with common rust disease					
Corn leaf with northern leaf blight disease					
Corn leaf with cercospora leaf spot/grey leaf spot disease					

TABLE 3: Potato plant leaf with disease (source: PlantVillage data set [22]).

Healthy leaf	Leaf with late blight disease
	
	
	
	
	

$$s(t) = (x * w)(t). \quad (2)$$

In convolutional neural network terminology, the first term (in this case (x)) is usually called the input to a stable operation, and the second argument (w in our case) is called

the kernel. The output or result of this rotation is called the feature map.

Discrete data can be used when working with a computer, so the integral functions of logical functions must be converted into a number of continuous “discrete” functions as follows:

TABLE 4: Tomato plant leaf with disease (source: PlantVillage data set [22]).








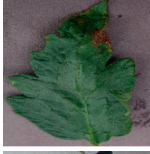






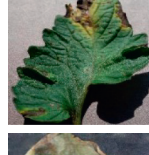

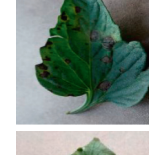
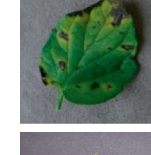





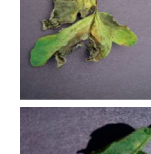








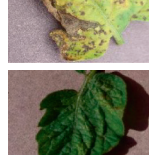



















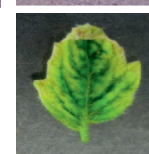
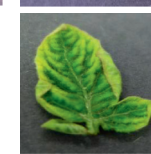






Healthy tomato leaf						
"Tomato leaf with bacterial spot disease"						
"Tomato leaf with early blight disease"						
"Tomato leaf with late blight disease"						
"Tomato leaf with mold disease"						
"Tomato leaf with septoria leaf spot disease"						
"Tomato leaf with two-spotted spider mite disease"						
"Tomato leaf with target spot disease"						
"Tomato leaf with mosaic virus disease"						
"Tomato yellow leaf curl virus disease"						

TABLE 5: Rice plant leaf with disease (source: PlantVillage data set [20]).

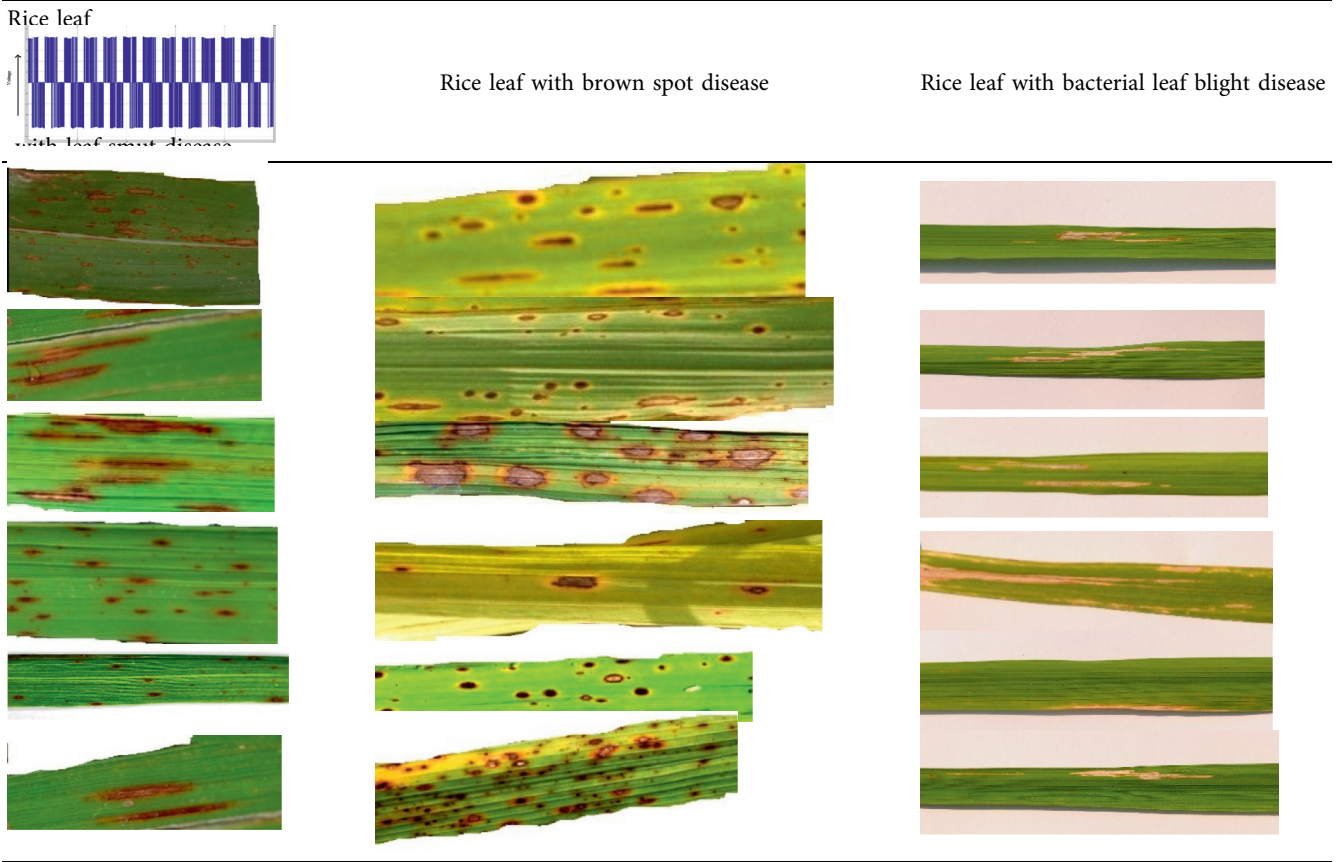


FIGURE 2: Typical deep convolutional neural network architecture [6].

$$S(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \quad (3)$$

In deep learning applications, the input is usually a multidimensional vector (tensor), and the kernel modified by the learning algorithm is usually a multidimensional parameter vector. For example, when image I is used as input, then a two-dimensional kernel denoted by K is often used:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i-m, j-n)K(m, n). \quad (4)$$

In practice, discrete convolutions can be viewed as multiplication by matrices:

Referring to Figure 3, we observe how 2D input I remains as the kernel of K , which is also a 2D array, and moves along input I performing element-wise multiplication and addition operations. Get the convolution result ($I \times K$).

3.2.3. Pooling. There are usually three layers in a CNN. At the first level of the network, the operation of the input data is folded. In the second step, all functions removed from the decontamination process are passed to an activation function; the most commonly used activation function is ReLU (reformed linear unit). This stage is sometimes called the perception stage [26].

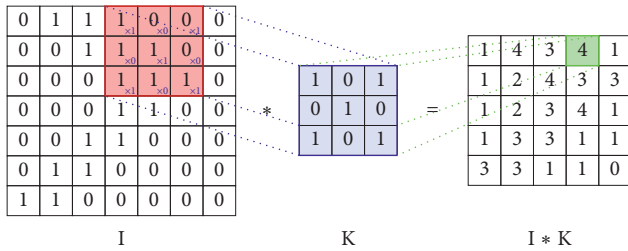


FIGURE 3: 2D convolution level function of a CNN [26].

In the last step, the join function is performed, which replaces the previous output or network output with a statistical summary generated by the previous neural network layer region. This is much easier to understand in Figure 4.

Figure 4 shows how the 2×2 zoning process is implemented. There are two types of pooling:

Max Pooling: this operation selects the maximum value of all parameters such that the attribute or value is reduced by a factor of 4.

Average Pooling: this operation selects the arithmetic mean of the region to use, resulting in a data reduction factor of 4.

3.2.4. Model, Architectures, and Committees. Considering the leading role of convolutional neural networks in various state of the art computer vision tasks [27], this was the model chosen to address the task in question, however, taking into account the different possible architectures for them. Trade-offs between the number of layers and trainable parameters, such as the computational costs required for training, must be assessed, particularly for those already established in the literature and with strong performance in computer vision applications.

In this context, for the scope of the present work, it was decided to use convolutional neural networks with fewer parameters when compared to the canonical solutions in the literature. Thus, the chosen architectures are as follows:

- (1) LeNet: initially proposed for the task of recognizing handwritten digits, this network consists of two convolutional layers followed by layers of max pooling in order to extract characteristics. Finally, a final convolutional layer is followed by two completely connected layers for classification of the outlet [28].
- (2) AlexNet: aiming at the use of convolutional neural network architecture with good performance reported in the related works, this network is composed of five initial convolutional layers and three layers completely connected at the end to produce the classification. It also has intermediate layers of dropout and max pooling [6].
- (3) MobileNet: for use on mobile and embedded devices, this convolutional neural network is based on deep separable convolution operations, which reduces the burden of operations to be carried out in the first layers [29].

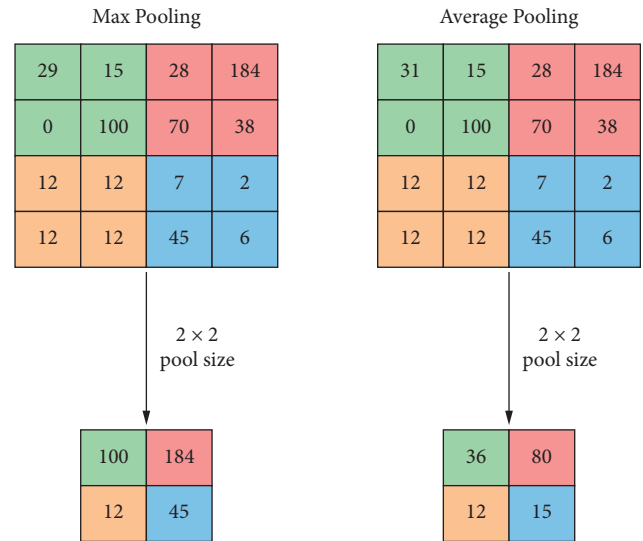


FIGURE 4: Pooling function with maximum clustering of left and middle right pooling [26].

- (4) ShuffleNet: it is based on two operations introduced by the authors: the so-called group convolutions, which are multiple convolutions in which each covers a portion of the input channels, and the shuffling of channels, which randomly mix the output channels of the convolutions in the group. According to its proponents, this architecture has a low computational cost while maintaining good accuracy [30].
- (5) EffNet: it resembles the MobileNet and ShuffleNet networks in terms of the use of in-depth separable convolution operations but introduces a new convolutional block that reduces the computational burden while exceeding the state of the art performance for some widely known databases [31].

Considering the previously mentioned architectures, we have the number of trainable, nontrainable, and total parameters as shown in Table 6. Note that the sum of the number of total parameters is lower than that of the VGG16 and VGG19 architectures [32], which suggest that their combination, according to a committee, may prove to be less costly than well-established architectures in the literature, under the terms considered. The considered architectures were trained according to the methodology previously described and evaluated individually in view of the performance in the test set.

3.3. Classification of Deep Features by Bayesian Optimized Support Vector Machine. In supervised learning, the classifier is trained through the presentation of a set of examples (input and desired output). A training set is used in supervised methods to educate the model to produce the desired output. This training dataset contains both right and incorrect outputs, allowing the model to improve over time.

It is expected that based on this knowledge, the classifier will be able to accurately predict the output of new data not previously presented, being able to act in a linear or nonlinear way.

TABLE 6: Parameters of the convolutional neural networks considered.

Architecture	Trainable parameters	Nontrainable parameters	Total parameters
LeNet	19,479,481	0	24,973,547
AlexNet	30,847,124	0	33,741,772
MobileNet	3,184,795	20,748	3,119,529
ShuffleNet	1,897,632	32,054	1,976,586
EffNet	604,552	2,033	619,832
Total	56,013,584	54,835	64,431,266

Considering the SVM, they divide the feature space into regions using an optimal separation hyperplane positioned exactly in the center between the margins of the two classes. Among the nonlinear functions that can be used in the SVM analysis are quadratic, polynomial, radial basis function (RBF), and Gaussian and two-layer perceptron.

This technique seeks to maximize the separation margin of samples from two groups. The solution to this optimization problem has a broad and established mathematical theory and can be expressed by the following equation:

$$\max W(\lambda) = \sum_{i=1}^N \lambda_i - \frac{1}{2} \sum_{i,j=1}^N y_i y_j \lambda_i \lambda_j (x_i x_j), \quad (5)$$

$$\text{subject to } \begin{cases} 0 \leq \lambda_i \leq C, \\ \sum_{i=1}^N \lambda_i y_i = 0, \quad i = 1, 2, \dots, N, \end{cases}$$

where C is the error limit, N is the number of samples, λ_i are the Lagrange multipliers, y is the desired output, and x are the input samples.

“Remember Bayes’ theorem. Suppose that A and B are two events for which the conditional probability $P(B|A)$ is known, then the probability $P(A|B)$ is defined as follows:”

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \quad (6)$$

where “ $P(A)$ is the prior probability; $P(B|A)$ is the probability of event B , depending on the occurrence of event A ; and $P(A|B)$ is the posterior probability.”

“Then some utility functions are maximized in the next model to determine the next point to be evaluated, and new observations are collected to repeat until the criterion stops.”

Since the SVM method uses sampling techniques for continuous parameters, it provides less accurate results. In the proposed work, an algorithm is analyzed that allows you to customize the SVM parameters. In research, sampling is extremely beneficial. It is one of the most essential aspects in determining how accurate your research/survey results are. If there is a problem with your sample, it will be reflected in the final result. Sampling techniques are of many types, such as sample random sampling and multistage sampling.

One of the directions of Bayesian optimization is the optimization of continuous variables and mixed (discrete and continuous) variables by solving problems with different data types. The main objective of using Bayesian optimization here is to find the suitable value for each parameter of SVM. There are at least three important practical choices that we need to

consider: the kernel functions, selection of its hyperparameters, and the acquisition functions. A default choice of covariance function is to use a squared exponential kernel [33].

$$K_{M52}(x, x') = \theta_0 \left(1 + \sqrt{5r^2(x, x')} + \frac{5}{3}r^2(x, x') \right) \exp \left\{ -\sqrt{5r^2(x, x')} \right\}. \quad (7)$$

The above kernel function itself has few parameters that need to be managed (such as covariance amplitude θ_0 and the observation noise ν). It can be done by marginalizing over hyperparameters and computing the integrated acquisition function.

3.4. Preprocessing for Second Phase. The previous subsections detailed the first part of the methodology, and the second phase starts from this section. Initially, the input image is resized into 300×450 pixel. This image is in RGB format so it will be converted from RGB to grey format to perform the texture and deep features extraction. Also, RGB to LAB conversion is used for the color features.

3.5. Color and Texture Feature Extraction

3.5.1. Color Moments. Moments of color are calculated to measure the brightness and intensity of an image. The color moments used in this study to remove color images are standard and standard deviation. The mean can be understood as the mean of the colors in the image, and the square root of the variance can be defined as the standard deviation. The histogram method uses color distribution. We need to store a lot of data. Instead of calculating the total distribution, only the dominant color attributes such as the mean and standard deviation are considered [34].

Moment 1:

$$\text{mean} = E_i = \sum_{j=1}^N \frac{1}{N} P_{ij}. \quad (8)$$

Moment 2:

$$\text{standard deviation} = \sigma_i = \sqrt{\left(\frac{1}{N} \sum_{j=1}^N (P_{ij} - E_i)^2 \right)}. \quad (9)$$

3.6. Gray-Level Cooccurrence Matrix (GLCM). The GLCM used for gray-level images shows the relationship between two neighboring pixels. It is determined based on the

distance and angle between pixels. The gray-level cooccurrence matrix (GLCM) also known as the gray-level spatially dependence matrix is a static approach to assessing texture that takes into account the spatial relationship of pixels. These parameters reveal info on images quality.

GLCM showing the spatial relationship of the image for the vector d is an N-dimensional square matrix that specifies the quantity of i - and j -valued pixel pairs [35]. Let's express a gray-level image with the function $I(r, c)$. Let $d = (d_r, d_c)$ be the spatial relation vector. The coformation matrix C_d is expressed as follows [35]:

$$C_d(i, j) = |\{(r, c): I(r, c) = i \text{ and } I(r + d_r, c + d_c) = j\}|. \quad (10)$$

Besides the distance between the two pixels, the orientation of the pixel pair is also important. These directions can

$$\text{energy} = \sum_i \sum_j N_d^2(i, j),$$

$$\text{contrast} = \sum_i \sum_j (i - j)^2 N_d(i, j), \text{homogeneity} = \sum_i \sum_j \frac{N_d(i, j)}{1 + |i - j|}, \text{correlation} = \frac{\sum_i \sum_j (i - \mu_i)(j - \mu_j) N_d(i, j)}{\sigma_i \sigma_j}, \quad (12)$$

where μ_i and μ_j are the arithmetic mean of the row and column sums of the gray-level cooccurrence matrix, respectively, and σ_i and σ_j are the standard deviation of the row and column totals, respectively.

Calculated energy, contrast, homogeneity, and correlation values are features of the image. The attribute vector of the image is created with these values.

4. HoG (Histograms of Oriented Gradient)

HoG is used for object detection in computer vision, and image processing counts the appearance of orientation gradients in localized portions of an image recognition window or region of interest (ROI). There are libraries that already implement this algorithm, for example, OpenCV HOG Descriptor. The implementation of the HoG algorithm looks like this, and it can be seen in Figure 6.

4.1. BPSO-Based Feature Selection. PSO is an understanding process. It is a complex algorithm that covers a particular herd of cattle. BPSO is a heuristic optimization approach that is commonly used to solve problems in continuous domains. It is a type of PSO that is applied to binary domains but uses continuously PSO's speed and inertia ideas, resulting in poor performance.

Algorithms for the development of particle clusters were developed by analyzing the behavior of birds, fish, and bees [38].

be $\theta = 0^\circ, 45^\circ, 90^\circ, 135^\circ$. In Figure 5, cooccurrence matrices in three different directions obtained from a 4×4 image are seen [35].

The normalized gray-level cogeneration matrix N_d and the symmetrical gray-level cogeneration matrix S_d are expressed as follows [36]:

$$N_d(i, j) = \frac{C_d(i, j)}{\sum_i \sum_j C_d(i, j)}, \quad (11)$$

$$S_d(i, j) = C_d(i, j) + C_{-d}(i, j).$$

By using the normalized gray-level cooccurrence matrix, the properties of the image including the texture characteristics such as energy, contrast, homogeneity, and correlation can be calculated. The specified properties are calculated with the following equations [36]:

Binary optimization was introduced [38]. If the problem can be eliminated or summarized, binary optimization can be helpful to resolve this unique problem [39]. Many optimization problems, such as conversion or rendering problems, are checked in a discrete space.

For the search area $S = \{0, 1\}^D$, the fitness function f maximizes, that is, $(\max f(x))$. The i^{th} particle in D dimension is defined as follows:

$$X_i = (x_{i1}, x_{i2}, \dots, x_{id})^T, x_{id} \in \{0, 1\}, \quad d = 1, 2, \dots, D, \\ V_i = (v_{i1}, v_{i2}, \dots, v_{id})^T, v_{id} \in [-V_{\max}, V_{\max}], \quad d = 1, 2, \dots, D, \quad (13)$$

where V_{\max} is the vector of the maximum velocity.

The best position at the moment can be characterized as follows [39]:

$$p_i = (p_{i1}, p_{i2}, \dots, p_{id})^T, p_{id} \in \{0, 1\}, \quad d = 1, 2, \dots, D. \quad (14)$$

The mathematical notation described above can also be defined by the following.

Equation for velocity:

$$v_{id} = v_{id} + c_1 \text{rand}_1(p_{id} + x_{id}) + c_2 \text{rand}_2(p_{gd} - x_{id}). \quad (15)$$

Equation for position:

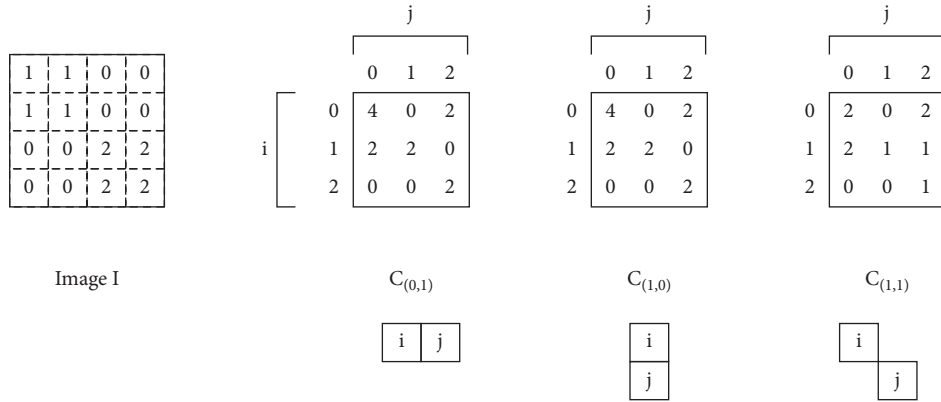


FIGURE 5: Cooccurrence matrices [35].

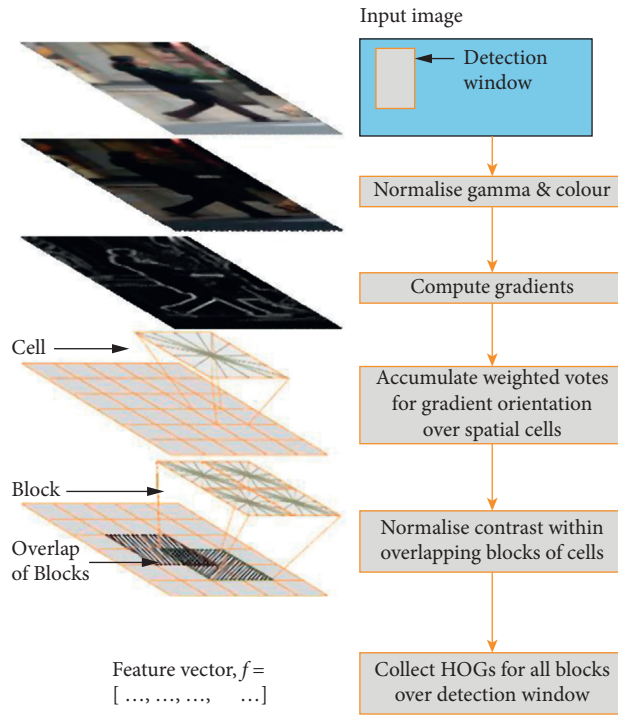


FIGURE 6: Procedure for HoG technique [37].

$$X_{i,d} = \begin{cases} 1 & \text{if } U(0,1) < \text{sigm}(v) \\ 0 & \text{otherwise} \end{cases}, \quad d = 1, 2, \dots, D; i = 1, 2, \dots, N. \quad (16)$$

Transfer function:

$$\text{sigm}(v_{i,d}): \frac{1}{1 + \exp(-\lambda v_{i,d})}, \quad (17)$$

where g is “index of the best performing particle”; p_{gd} is “best part”; N is “the width of the fortification”; c_1, c_2 are

“social and cognitive component constants”; $\text{rand}_1, \text{rand}_2$ are $U(0,1)$ random numbers; $\text{sigm}(v_{i,d})$ is sigmoid transform function

4.2. Classification by Random Forest Classifier. A random forest is a classifier that consists of a collection of classification trees $\{h(x, T, \Theta_k), k = 1, 2, \dots, K\}$ where Θ_k represents vectors identically and independently distributed. Each tree in the collection (forest) expresses only one vote to assign the statistical unit to a class on the

basis of the vector of values x : the final choice is to attribute the statistical unit to the class for which the majority of votes were obtained, that is, for which the majority of the trees of the random forest have expressed themselves [40].

The classification based on random forests has very interesting statistical characteristics:

- (i) It is relatively robust with respect to extreme observations (outliers) and experimental noise
- (ii) It is faster than many other numerical classification procedures
- (iii) It allows internal estimates of the error, correlation, and importance of the variables used in the classification process
- (iv) It is relatively simple and can be implemented on parallel computers efficiently
- (v) It can be easily parallelized

One of the fundamental points that characterize random forests is that the generalization error converges “almost certainly” for a number of trees in the forest that diverge, and therefore, the possibility of overfitting the overall classification procedure is avoided due to the increase in the number of trees.

5. Simulation Results

5.1. Evaluation Parameters. The evaluation parameters for defining the efficiency of the model are shown in Table 7.

$$\begin{aligned}
 \text{accuracy} &= \frac{TP + TN}{TP + TN + FP + FN}, \\
 \text{precision} &= \frac{TP}{TP + FP}, \\
 \text{sensitivity} &= \frac{TP}{TP + FN}, \\
 F - \text{score} &= \frac{2TP}{2TP + FP + FN}.
 \end{aligned} \tag{18}$$

5.2. Results and Discussion. This section investigates the role of different classification techniques. These techniques are based on a convolutional neural network and a Bayesian optimized support vector machine. Both techniques play a crucial role in the detection of plant diseases. Figures 7–12 show the step-wise procedure followed by the methodology applied.

The confusion matrix plot of proposed plant leaf detection using Bayesian optimized support vector machine classifier and random forest classifier is shown in Figures 13 and 14, respectively.

Here, $TP = 58$, $TN = 225$, $FP = 3$, and $FN = 21$.

$$\begin{aligned}
 \text{accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} = \frac{58 + 225}{58 + 225 + 3 + 21} = 92.2\%, \\
 \text{precision} &= \frac{TP}{TP + FP} = \frac{58}{58 + 3} = 95.1\%, \\
 \text{sensitivity} &= \frac{TP}{TP + FN} = \frac{58}{58 + 21} = 73.4\%, \\
 F - \text{score} &= \frac{2TP}{2TP + FP + FN} = \frac{2 \times 58}{2 \times 58 + 3 + 21} = 82.85\%.
 \end{aligned} \tag{19}$$

Here, $TP = 70$, $TN = 225$, $FP = 3$, and $FN = 9$.

$$\begin{aligned}
 \text{accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} = \frac{70 + 225}{70 + 225 + 3 + 9} = 96.1\%, \\
 \text{precision} &= \frac{TP}{TP + FP} = \frac{70}{70 + 3} = 95.9\%, \\
 \text{sensitivity} &= \frac{TP}{TP + FN} = \frac{70}{70 + 9} = 88.6\%, \\
 F - \text{score} &= \frac{2TP}{2TP + FP + FN} = \frac{2 \times 70}{2 \times 70 + 3 + 9} = 92.1\%.
 \end{aligned} \tag{20}$$

The comparison for the efficiency parameters for the proposed methodology and previously used methodology is shown in Table 8.

6. Discussion

The advancement in deep learning (DL) presents an opportunity to extend research and application based on the identification of plant diseases using digital images. Fast and accurate models are required so that the right measures can be applied early. The network architecture chosen for the creation of a classifier system depends on whether the goal is to maximise or minimise. On the one hand, if the problem requires constant reconfiguration the SqueezeNet network is the fastest; however it is too unstable to model. If high ranking accuracy is desired, there are alternatives such as AlexNet and GoogLeNet. In the study carried out by Brahim and others [14], it was based on classifying diseases for some CNNs. Performing the small comparison based on accuracy and using the same learning transfer technique, InceptionV3 was the one with the lowest percentage of accuracy obtained; in the same way, AlexNet did not achieve similar results because AlexNet was the network with the best results achieved while scored well below what the authors mention. Notably, model training takes too many hours on a high-performance GPU computing. Finally, the PlantVillage data set is not balanced since some classes have more images than others, which could be a downside and result in overfitting if not trained correctly.

Deep learning has achieved great results in many fields of research due to the great ability to form characteristics in a fully automated way without the intervention of humans. In

TABLE 7: Evaluation parameters.

TP (true positive)	“Indicated the number of plant leaf diseases that were classified as correctly classified”
TN (true negative)	“Indicated the number of plant leaf diseases that were classified as not classified correctly”
FP (false positive)	“Indicated the number of plant leaf diseases that were classified as incorrectly classified”
FN (false negative)	“Indicated the number of plant leaf diseases that were classified as not classified incorrectly”



FIGURE 7: Input image.

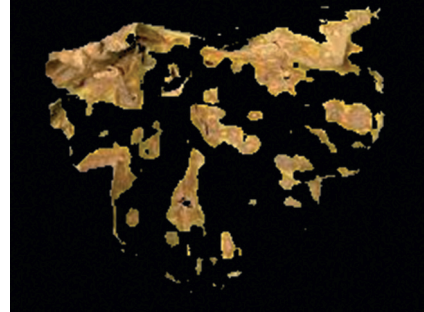


FIGURE 11: Objects in cluster 3.



FIGURE 8: Image labeled by cluster index.

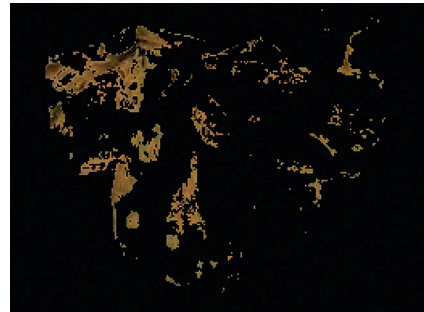


FIGURE 12: Blue nuclei.

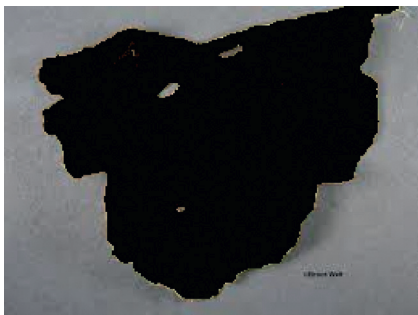


FIGURE 9: Objects in cluster 1.

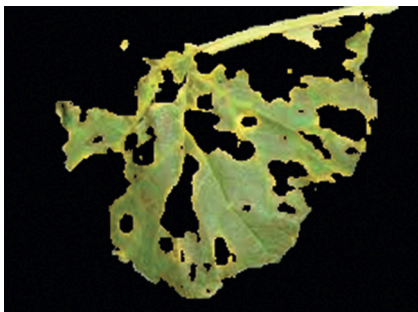


FIGURE 10: Objects in cluster 2.

	58 18.9%	3 1.0%	95.1% 4.9%
Output Class	21 6.8%	225 73.3%	91.5% 8.5%
	73.4% 26.6%	98.7% 1.3%	92.2% 7.8%
	Target Class		

FIGURE 13: Confusion matrix plot of proposed plant leaf detection using Bayesian optimized support vector machine classifier.

the protection of plant diseases, many works have proposed the use of DL to detect and classify diseases that is why we have proposed the use of CNN with the aim of creating a tool

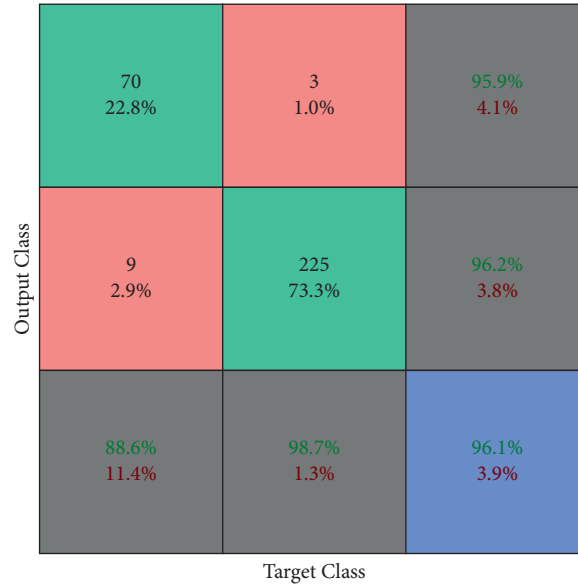


FIGURE 14: Confusion matrix plot of proposed plant leaf detection using random forest classifier.

TABLE 8: Comparative results.

Methods	Method used	Accuracy	Precision	Sensitivity	F1-score
Ahmad et al. (2020) [41]	Tomato plant leaf disease detection from PlantVillage database using CNN	—	84.5%	90.6%	87.6%
Kaur et al. (2021) [42]	Bell pepper, potato, and tomato plant leaf disease detection from PlantVillage database using LBP, GLCM, SIFT, and Gabor features with ensemble classifier	95.66%	—	—	—
Rao et al. (2020) [43]	Various plants leaf disease detection from PlantVillage database using GLCM, complex Gabor filter, curvelet, and image moments features with neuro-fuzzy logic classifier	93.18%	—	—	—
Zong et al. (2020) [44]	Apple leaf disease detection from PlantVillage database using DenseNet-121 deep convolution network	93.71%	—	—	—
Sujatha et al. (2021) [45]	Various plants leaf disease detection from PlantVillage database using deep convolution network (VGG-16)	89.5%	—	—	—
Chen et al. (2020) [46]	Rice plant leaf disease detection from Plant Village database using dense net pretrained on ImageNet	94.07%	—	—	—
Inception module of convolutional neural network		94.07%	—	—	—
Proposed Bayesian optimized SVM	Various plants leaf disease detection from PlantVillage database using deep convolution network and Bayesian optimized SVM classifier	92.2%	95.1%	73.4%	82.85%
Random forest classifier	Various plants leaf disease detection from PlantVillage database using hybrid features and random forest classifier	96.1%	95.9%	88.6%	92.1%

for those researchers who need to design and implement a classification automation of plant leaf diseases, giving you the precise data on the architecture that best suits you. On the other hand, ResNet50 is computationally more expensive in terms of execution time. Furthermore, ResNet50, ResNet101, and InceptionV3 being the deepest networks show that they were not as accurate despite a large number of depth layers. Finally, it was proposed to analyze different CNN in time and performance. AlexNet managed to obtain a better result, while ShuffleNet achieved it in less time being below 1.53%. Likewise, in our study, we make use of the

activation layers of AlexNet in order to detect and locate the regions of the disease.

7. Conclusion

The convolutional neural networks represent a deep learning architecture that has been achieving remarkable prominence in image recognition. Five convolutional neural network architectures were trained and tested for the problem in question, to mention: LeNet, ShuffleNet, AlexNet, EffNet, and MobileNet, the latter having achieved better

performance among them, with an accuracy of 96.1%. All networks were combined in committees subject to three voting strategies, by the majority, mediated by hybrid feature-based random forest and mediated by Bayesian optimized SVM. In addition to the good performance for the proposed task, it was observed that the total of trainable parameters of the best committee was lower than the canonical architectures VGG16 and VGG19, widely used in computer vision tasks. In general, the committees proposed for the tasks performed well in the testing stage, which leads to good potential for use in practical scenarios under similar conditions. The related works in the literature for the same database are considered a multilabel classification, which makes comparative performance analysis difficult with the results proposed here. Despite this difficulty, it is emphasized that approaching the problem as a binary classification task can facilitate the use and adaptation of the model proposed here for other contexts, reducing the burden of human specialists in annotating examples, which can be particularly useful by means of other plant images of interest and different types of diseases that may affect them. The scope of this research is that it proves CNNs' technical capability in diagnosing plant diseases and paves the way for artificial intelligence solutions for farmers. This can be particularly useful in the practical context of agriculture, where many assessments of this nature need to be carried out in the field. To compare the results of the convolutional neural network, the Bayesian optimized support vector machine and hybrid feature-based random forest classifier were used with a collection of features extractors of color and texture. As a result, using convolutional neural networks, this work achieved a maximum accuracy of 96.1% in the detection of leaf diseases of apple, corn, potato, tomato, and rice plants. [47]

Data Availability

The data shall be made available on request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research work is self-funded.

References

- [1] R. N. Strange and P. R. Scott, "Plant disease: a threat to global food security," *Annual Review of Phytopathology*, vol. 43, pp. 83–116, 2005.
- [2] A. K. Rangarajan, R. Purushothaman, and A. Ramesh, "Tomato crop disease classification using pre-trained deep learning algorithm," *Procedia Computer Science*, vol. 133, pp. 1040–1047, 2018.
- [3] F. W. Nutter, P. D. Esker, and R. A. C. Netto, "Disease assessment concepts and the advancements made in improving the accuracy and precision of plant disease data," *European Journal of Plant Pathology*, vol. 115, no. 1, pp. 95–103, 2006.
- [4] J. G. Arnal Barbedo, "Plant disease identification from individual lesions and spots using deep learning," *Biosystems Engineering*, vol. 180, pp. 96–107, 2019.
- [5] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers of Plant Science*, vol. 71419 pages, 2016.
- [6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.
- [7] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, Boston, MA, June 2015.
- [8] L. Quan, H. Feng, Y. Lv et al., "Maize seedling detection under different growth stages and complex field environments based on an improved Faster R-CNN," *Biosystems Engineering*, vol. 184, pp. 1–23, 2019.
- [9] Y. Yu, K. Zhang, L. Yang, and D. Zhang, "Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN," *Computers and Electronics in Agriculture*, vol. 163, Article ID 104846, 2019.
- [10] A. Fuentes, S. Yoon, S. Kim, and D. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," *Sensors*, vol. 17, no. 9, 2022 pages, 2017.
- [11] I. M. Hanssen and M. Lapidot, "Major tomato viruses in the Mediterranean basin," *Advances in Virus Research*, Academic Press, vol. 84, , pp. 31–66, 2012.
- [12] G. Thakre, A. R. More, and K. S. Gajakosh, "A study on real time plant disease diagnosis system," *International Journal-ofAdvanceResearch, IdeasandInnovationsinTechnology*, vol. 3, pp. 1118–1124, 2017.
- [13] J. G. A. Barbedo, "Factors influencing the use of deep learning for plant disease recognition," *Biosystems Engineering*, vol. 172, pp. 84–91, 2018.
- [14] M. Brahim, 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.
- [15] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, *Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification*, Computational intelligence and neuroscience, 2016.
- [16] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," *Neurocomputing*, vol. 267, pp. 378–384, 2017.
- [17] Y. Kawasaki, H. Uga, S. Kagiwada, and H. Iyatomi, "Basic study of automated diagnosis of viral plant diseases using convolutional neural networks," *Advances in Visual Computing*, Springer, in *Proceedings of the International Symposium on Visual Computing*, pp. 638–645, October 2015.
- [18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, NV, USA, December 2016.
- [19] F. N. Iandola, S. Han, M. W. Mickiewicz, K. Ashraf, W. J. Dally, and K. Kreutzer, "Squeeze Net: Alex Net-level accuracy with 50x fewer parameters and < 0.5 MB model size," in *Proceedings of the International Conference on Learning Representations*, Toulon, France, April 2016.
- [20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference*

- on computer vision and pattern recognition, pp. 770–778, Las Vegas, NV, USA, June, 2016.
- [21] D. Hughes and M. Salathé, “An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing,” 2015, <https://arxiv.org/abs/1511.08060>.
- [22] D. R. Hammou and M. Boubaker, “Tomato plant disease detection and classification using convolutional neural network architectures technologies,” in *Networking, Intelligent Systems and Security*, pp. 33–44, Springer, Singapore, 2022.
- [23] R. Khan, N. Tyagi, and N. Chauhan, “Safety of food and food warehouse using VIBHISHAN,” in *Journal of Food Quality*, A. Durazzo, Ed., vol. 2021, Hindawi Limited, Journal of Food Quality, , pp. 1–12, 2021.
- [24] Y. Toda and F. Okura, *How Convolutional Neural Networks Diagnose Plant Disease*, Plant Phenome’s, Berlin, Article ID 9237136, 2019.
- [25] T. Wiatowski and H. Bölcskei, “A mathematical theory of deep convolutional neural networks for feature extraction,” *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1845–1866, 2017.
- [26] P. Taraba, “Linear regression on a set of selected templates from a pool of randomly generated templates,” *Machine Learning with Applications*, vol. 6, Article ID 100126, 2013.
- [27] S. Khan, H. Rahmani, S. A. A. Shah, and M. Bennamoun, “A guide to convolutional neural networks for computer vision,” *Synthesis Lectures on Computer Vision*, vol. 8, no. 1, pp. 1–207, 2018.
- [28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [29] R. Khan, S. Kumar, N. Dhingra, and N. Bhati, “The use of different image recognition techniques in food safety: a study,” in *Journal of Food Quality*, I. Tomasevic, Ed., vol. 2021, Hindawi Limited, Journal of Food Quality, , pp. 1–10, 2021.
- [30] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shuffle net: an extremely efficient convolutional neural network for mobile devices,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, Nashville, USA, June 2018.
- [31] I. Freeman, L. Roese-Koerner, and A. Kummert, “October. Eff net: an efficient structure for convolutional neural networks,” in *Proceedings of the 25th IEEE International Conference on Image Processing (ICIP)*, pp. 6–10, IEEE, Athens, Greece, October 2018.
- [32] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Proceedings of the International conference learning representation*, San Diego, CA, USA, 2015.
- [33] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian optimization of machine learning algorithms,” *Advances in Neural Information Processing Systems*, vol. 25, pp. 2951–2959, 2012.
- [34] N. Keen, *Color Moments*, pp. 3–6, School Of Informatics, University Of Edinburgh, Edinburgh, 2005.
- [35] K. Khairnar and R. Dagade, “Disease detection and diagnosis on plant using image processing A review,” *International Journal of Computer Application*, vol. 108, no. 13, pp. 36–38, 2014.
- [36] R. M. Prakash, G. P. Saraswathy, G. Ramalakshmi, K. H. Mangaleswari, and T. Kaviya, “Detection of leaf diseases and classification using digital image processing,” in *Proceedings of the 2017 international conference on innovations in information, embedded and communication systems (ICI-IECS)*, pp. 1–4, IEEE, Tamil nadu, India, 2017 March.
- [37] H. Rizk, “Automated early plant disease detection and grading system: development and implementation,” Available online at: <https://fount.aucegypt.edu/cgi/viewcontent.cgi?article=1173&context=etds>, 2017.
- [38] J. Kennedy and R. C. Eberhart, “A discrete binary version of the particle swarm optimization,” *Computational cybernetics and simulation*, vol. 5, pp. 4104–4108, 1997.
- [39] M. A. Khanesar, M. Teshnehlab, and M. A. Shoorehdeli, “June. A novel binary particle swarm optimization,” in *Proceedings of the 2007 Mediterranean conference on control & automation*, pp. 1–6, IEEE, Athens, Greece, June 2007.
- [40] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [41] I. Ahmad, M. Hamid, S. Yousaf, S. T. Shah, and M. O. Ahmad, “Optimizing retrained convolutional neural networks for tomato leaf disease detection,” *Complexity*, vol. 2020, 2020.
- [42] N. Kaur, “Plant leaf disease detection using ensemble classification and feature extraction,” *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 11, pp. 2339–2352, 2021.
- [43] A. Rao and S. B. Kulkarni, “A hybrid approach for plant leaf disease detection and classification using digital image processing methods,” *International Journal of Electrical Engineering Education*, Article ID 0020720920953126, 2020.
- [44] Y. Zhong and M. Zhao, “Research on deep learning in apple leaf disease recognition,” *Computers and Electronics in Agriculture*, vol. 168, Article ID 105146, 2020.
- [45] R. Sujatha, J. M. Chatterjee, N. Jhanjhi, and S. N. Brohi, “Performance of deep learning vs machine learning in plant leaf disease detection,” *Microprocessors and Microsystems*, vol. 80, p. 103615, 2021.
- [46] J. Chen, D. Zhang, Y. A. Nanekaran, and D. Li, “Detection of rice plant diseases based on deep transfer learning,” *Journal of the Science of Food and Agriculture*, vol. 100, no. 7, pp. 3246–3256, 2020.
- [47] M. Hussein and A. H. Abbas, “Plant leaf disease detection using support vector machine,” *Al-Mustansiriyah Journal of Science*, vol. 30, no. 1, pp. 105–110, 2019.