Multimedia Tools and Applications (2025) 84:1697–1724
https://doi.org/10.1007/s11042-024-18991-6

An ensembled-deep-learning paradigm trained with a self-improved coyote optimization algorithm (SI-COA) for crop disease detection

Preeti Shukla[1] · Amit Kumar Chandanan[2]

Received: 21 March 2023 / Revised: 28 February 2024 / Accepted: 14 March 2024 /
Published online: 17 April 2024

Abstract
In recent times, drastic climate changes have caused a substantial increase in the growth of crop diseases. This causes large-scale demolition of crops, decreases cultivation, and eventually leads to the financial loss of farmers. Due to the rapid growth in a variety of diseases and adequate knowledge of farmers, identification, and treatment of the disease have become a major challenge. The leaves have texture and visual similarities which are attributes for the identification of disease type. Hence, computer vision employed with deep learning provides a way to solve this problem. In this research work, a novel voting-based ensembled-deep-learning paradigm is developed for crop disease detection. Initially, the collected raw images are pre-processed via Laplacian Filter and morphological operations. From, the pre-processed image, the ROI region is identified using K-means Clustering. Then, from the ROI regions, the features like texture feature (Chi-squared Local Binary Pattern (CLBP)), Scale Invariant Feature Transform (SIFT), Gray-Level Co-occurrence Matrix (GLCM)), color features, and shape features are extracted. Then, the optimal features are selected by using Self-Improved Coyote Optimization Algorithm (SI-COA) is an extended version of the standard Coyote Optimization Algorithm (COA) and is a metaheuristic for optimization that is inspired by the canis latrans species. For crop disease detection, the voting-based ensembled-deep-learning paradigm was used which includes the Attention-based Bi-LSTM, Recurrent Neural Networks (RNNs), and Optimized Deep Neural Networks (O-DNN). The weight of DNN is optimized via Self-Improved Coyote Optimization Algorithm (SI-COA). The overall outcome of this process is evaluated by using various performance metrics such as Precision, Recall, F1-Score, Accuracy, Sensitivity and Specificity, NPV, FNR, FPR, and MCC, as well.

Keywords Crop disease detection · Improved binary gabor pattern · SIFT · GLCM · Improved coyote optimization algorithm · COA · RNN

✉ Preeti Shukla
preeti.shuklaggu@gmail.com

[1] Department of Computer Science and Information Technology, Guru Ghasidas Vishwavidyalaya, Bilaspur, CG, India

[2] Department of Computer Science & Engineering, Guru Ghasidas Vishwavidyalaya, Bilaspur, CG, India

1 Introduction

Crop production yield per unit area increased as a result of the industrial and green revolutions, which met the food needs of the expanding population. However, they also led to an increase in the use of synthetic fertilizers in agriculture. The demand for plant products rises more dramatically each year as the Earth's population rises [1]. The ability to protect crops from plant diseases is essential for satisfying the growing demand for food, both in terms of quantity and quality [2]. Food insecurity, which is a significant cause of plant diseases, is one of the main problems affecting humanity today. According to one study, plant diseases are to blame for 16% of the global crop yield losses. Pest losses are anticipated to be between 26 and 29 percent for soybeans and about 50 percent for wheat globally [3]. Fungi, fungus-like species, bacteria, viruses, viroid, virus-like creatures, nematodes, protozoa, algae, and parasitic plants make up the major families of plant pathogens [4]. Economically speaking, plant diseases have an annual cost to the global economy of about US$220 billion. According to the Indian Council of Agricultural Research, pests, and diseases annually reduce crop productivity by more than 35%. As a result, the alarming increase in pests and diseases threatens food security. They have had a significant impact on social, economic, and environmental activities [5].

For those in agriculture, accurately identifying plant diseases is still essential [6]. Instead of asking other farmers for advice, they do not take advantage of multiple opportunities [7]. In some cases, it's also necessary to set up a lab to find the damaged leaves. However, in a small number of countries, farmers may not have access to the necessary tools or even expert assistance. Additionally, using an expert's services is expensive and time-consuming. As a result, it is advised to develop a novel technique for efficiently monitoring vast crop fields. It is discovered that automated disease detection using visual cues from plant leaves is easy and affordable. The idea of precise plant disease detection can be enhanced and broadened, and computer vision applications in precision agriculture can be created as a result of developments in this area [8, 9]. To find and categorize plant diseases, general image processing techniques like threshold and color analysis are used [10].

Recent advancements in computer vision technologies for precision agriculture have made the methodology for disease identification a crucial component of acquiring data about crop health monitoring. This has greatly increased the efficiency of disease detection and agricultural production output [11]. Early plant disease identification and prevention are essential to crop harvesting because they can effectively eliminate any growth problems and, as a result, reduce the need for pesticide use that causes pollution. cultivation of crops without [12]. The development of machine learning has greatly aided computer vision applications. Success results in the adoption of novel techniques and models that now belong to a new class called "Deep Learning" (DL). Deep learning is a computational approach that employs neural networks including numerous layers to acquire knowledge of patterns and features inherent in the input data. Deep learning models are employed within this particular framework to undertake the processing and analysis of data pertaining to crops and crop diseases. The identification process is most likely derived from the examination and evaluation of many characteristics of crops, such as pictures and sensor data, through the utilisation of ensembled deep learning models [13–15]. Due to their robustness, DL techniques have been used extensively in the agricultural sector [16]. The major contribution of this research work is:

- To introduce a new CLBP for extracting the texture features from the pre-processed images
- Select the optimal features using the new SI-COA, which is an extended version of the standard COA.
- To design a new voting-based ensembled-deep-learning paradigm with Attention-based Bi-LSTM, RNNs, and O-DNN.
- The weight of DNN is optimized via SI-COA.

2 Literature review

Ferentinos et al. [17] created Using convolutional neural networks and deep learning methods, it is possible to identify and classify plant illnesses using brief photos of healthy and diseased leaves in 2018. The models were trained using 87,848 photos from a public collection that included 25 distinct plant species in 58 different classes of pairings, including healthy plants. Following this, in 2019, Barbedo et al. [18] investigated the process accomplished by focusing on certain lesions and areas rather than scanning the entire leaf. Without using additional photos, the data's variability could be increased because each region had distinctive qualities. It makes it possible to identify various illnesses that affect the same leaf. However, effective symptom segmentation must be carried out manually, preventing full automation. In 2020, Guo et al. [19] provided a deep learning-based mathematical model that improves precision, generality, and training efficiency for recognizing and detecting plant diseases. The Region Proposal Network (RPN) was utilized to first identify and localize the leaves in a complex environment. Images that have been segmented based on the result of the RPN method then contain the feature of symptoms when using the Chan-Vese (CV) algorithm. Using the dataset of diseased leaves on a plain background, the segmented leaves were then added to the transfer learning model and trained. The model was also examined for resistance to the illnesses of rust, black rot, and bacterial plaque.

Abbas et al. [20] proposed a deep learning-based approach for detecting tomato diseases that creates artificial images of tomato plant leaves using the Conditional Generative Adversarial Network in 2021. The images of tomato leaves were classified into ten categories of diseases using a DenseNet121 model that used transfer learning to train on both fake and real images. The suggested model has undergone rigorous training and testing using the publicly available Plant Village dataset. The suggested methodology showed how it is superior to the existing methods. Succeeding this, in 2021, Zhao et al. [21] developed a Double GAN for growing plant leaves to expand the data set and evaluate the value of the produced leaves by classification accuracy. A lot of generative adversarial networks are used in the field of image generation (GAN). However, several GANs proposed by researchers are mainly employed to generate images utilizing sufficient sample sizes. In the database of plant diseases, unhealthy leaves were not frequently found. In 2021, Hassan et al. [22] suggested the diseases shallow VGG with RF and shallow VGG with XGBoost can be detected using two different methods. The suggested model was compared to various deep learning-based and manually constructed approaches. The tests included three distinct plants: maize, potatoes, and tomatoes. Infections that impact maize include blight, common rust, and grey leaf spot; infected potatoes include early and late blight; infected tomatoes include bacterial spots, early and late blight. The results showed that, in terms of accuracy, precision, recall, f1-score, and specificity, implemented shallow VGG with the XGBoost model outperforms several deep learning models. In 2021, Chen et al. [23]

adopted the MobileNet-V2 pre-trained on ImageNet as the backbone network and incorporated the attention mechanism to understand the importance of inter-channel relationships and spatial spots for input information. This increased the learning capability for minute lesion features. In the interim, the loss function was optimized and the transfer learning was repeated twice for the model training.

Later period of time, in 2022, Pandian et al. [24] suggested using photos of leaves, and a new 14-layered deep convolutional neural network (14-DCNN) was developed to recognize plant leaf diseases. A new dataset was created by fusing several open datasets. Using data augmentation techniques, the dataset's various class sizes were balanced. The three image augmentation techniques used were Neural Style Transfer (NST), Deep Convolutional Generative Adversarial Network (DCGAN), and Basic Image Manipulation (BIM). The collection includes 147,500 photos of 58 different classes of plant leaves, including classes with healthy and sick leaves as well as one class without leaves. The proposed DCNN model was trained using a Multi-Graphics Processing Unit (MGPU) environment for 1000 iterations. Next, Hassan et al. [25] presented a unique deep-learning model based on the residual connection and inception layer in 2022. With the use of depth-wise separable convolution, the number of parameters was reduced. The suggested model was trained on three different datasets for plant diseases and put to the test. The suggested model delivers superior accuracy with fewer parameters when compared to the most sophisticated deep learning models. In 2022, Pandian et al. [26] suggested a Deep Convolutional Neural Network (DCNN) model for image-based plant leaf disease diagnosis that makes use of data augmentation and hyperparameter tuning techniques. The enriched dataset used to train the DCNN model included more than 240,000 photographs of various healthy and ill plant leaves as well as background images. The five key image augmentation methods were applied: Position Augmentation, Colour Augmentation, Principal Component Analysis, and Generative Adversarial Network. The proposed DCNN model's hyperparameters were optimized using the random search technique. Show the given below Review of the existing works Table 1.

2.1 Research gaps

See Table 1.

3 Proposed methodology

In recent times, drastic climate changes and lack of immunity in crops have caused a substantial increase in the growth of crop diseases. This causes large-scale demolition of crops, decreases cultivation, and eventually leads to the financial loss of farmers. Due to the rapid growth in a variety of diseases and inadequate knowledge of farmers, identification, and treatment of the disease have become a major challenge. The leaves have texture and visual similarities which are attributes for the identification of disease type. Hence, computer vision employed with deep learning provides a way to solve this problem. This paper proposes a deep learning-based model for crop disease detection. The projected crop disease detection model includes five major phases: (a) pre-processing, (b) segmentation, (c) feature extraction (d) optimal feature selection, and (e) crop disease detection.

Table 1 Review of the existing works

Author	Aim/ Process	Research gaps
Chowdhury et al. [3]	Deep Learning-Based Automatic and Reliable Leaf Disease Detection	Diverse environments are not taken into account
Roy et al. [12]	A Deep Learning-Based Multi-Class Plant Disease Detection Model Based on Computer Vision	Processes for generic disease detection, automated agricultural detection, and the detection of different fruits and crops are not expanded
Abbas et al. [20]	Utilizing C-GAN synthetic images and transfer learning for tomato plant disease detection	It is not explained how to categorize or identify diseases in different plant parts, such as fruits, stems, and branches
Zhao et al. [21]	Using Double GAN-Based Generated Leaves to Detect Plant Disease	more detailed images are not generated
Hassan et al. [22]	Identification of Plant Disease Using a Shallow Convolutional Neural Network	It is not implemented to use portable smart devices for automating plant disease identification
Chen et al. [23]	Use of lightweight attention networks for disease detection in rice plants	It is not implemented to automatically monitor and detect the wide range of diseases affecting rice plants
Pandian et al. [24]	Utilizing Deep Convolutional Neural Networks to Detect Plant Disease	It is not possible to estimate the likelihood of plant disease or to assess its severity using deep learning

Dataset Images of both healthy and diseased plant leaves from the Plant Village [27] dataset acquired 54,309 photos taken at experimental research stations associated with Land Grant Universities in the USA are included in the data entry. The images show 14 different crop species, including Tomatoes, Pepper, Potato, and Apples. It includes images of 17 fungal, four bacterial, two Mold (oomycete), two viral, and one mite-related diseases. For the training of our proposed approach, a total of 20,637 images of the plant leaves of Potato, Pepper, and Tomato only are considered for evaluation purposes.

Collection process The team of experts would remove the leaves off the plant and gather them then, the leaves were positioned against a sheet of paper with a black or grey background. Using a regular point-and-shoot camera set to automatic mode, captures 4–7 pictures of each leaf (Sony DSC—Rx100/13 20.2 megapixels) [27].

Pre-processing The size of $256 \times 256 \times 3$ raw photos that were collected are pre-processed using morphological operations and a Laplacian filter to produce a 256×256 sized pre-processed image.

Feature extraction Then, from the pre-processed images, the features like texture feature (CLBP, SIFT, GLCM), color features (Color Histogram), shape features (Center of gravity, Orientation, Thickness, Anisotropy) were extracted.

Optimal feature selection Among the extracted features, the optimal features are selected using a self-improved meta-heuristic optimization model referred to as SI-COA. This SI-COA model is an extended version of the standard COA. This COA is a metaheuristic for optimization that is inspired by the canis latrans species.

3.1 Preprocessing

In this research, morphological procedures and the Laplacian Filter are used for pre-processing. The collected raw images are first pre-processed via Laplacian Filter, and then the contrast of the images is increased via morphological operations.

3.1.1 Laplacian filter

A second-order or second-derivative way of enhancing the image is the Laplacian, which draws attention to areas of rapid intensity shift in an image. Finding a photograph's minute details is one of its specialties. An enhancement using a Laplacian operator is made to any feature with an abrupt discontinuity. Known for approximating the second derivative of the expression given by, the Laplacian is a linear differential operator. These operations serve to augment the caliber and emphasize significant boundaries and intricacies within the visual representations, thereby rendering subsequent examination more efficacious.

$$\nabla^2 h = \frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} \tag{1}$$

As per Eq. (1) the term h denotes the image.

3.1.2 Morphological operations

In the context of ensemble deep learning, morphological operations refer to the application of mathematical morphological image processing methods to enhance the performance of deep learning models. These operations typically involve the manipulation of the shapes in an image, such as dilation, erosion, opening, and closing. They can be applied as either post-processing procedures to improve the output of a deep learning model or as pre-processing activities to improve the quality of the input data. By reducing noise and emphasizing key features in the input, morphological operations are used in ensemble deep learning to enhance the robustness and accuracy of the model. The morphological operations are explained as follows:

Dilation Dilation is a morphological image processing method that makes objects in an image larger. Utilizing a structuring element (also referred to as a kernel or filter), which establishes the shape of the dilation operation, it is carried out. Each pixel in the input image is changed during the dilation operation to its maximum value within the neighborhood specified by the structuring element. The size of structuring element's size and shape has an impact on how much the items in the output image have expanded in size. Dilation is widely employed in image processing applications for edge identification, image segmentation, and object recognition.

Morphological dilatation fills up small gaps in objects and increases object visibility. Shapes with filling appear larger and lines appear thicker. The minimum value of all the pixels in the neighborhood makes up the value of the output pixel. A pixel in a binary image is set to 0 if any of its neighbors also have a value of 0.

For sets Z and Y in $A2$ (Binary Image), dilation of Z by Y is denoted by $Z \oplus Y$.

$$Z \oplus Y = \{a|(Y^{\wedge})a \cap Z \neq \emptyset\} \tag{2}$$

Erosion Erosion is a morphological image processing operation that involves shrinking the boundaries of objects in an image. This is done by convolving a small structuring element with the image and replacing the center pixel of the structuring element with the minimum pixel value within the neighborhood defined by the structuring element.

Erosion (usually represented by $\ominus$) is one of the two fundamental operations in morphological image processing, along with dilation, upon which all other morphological operations are built.

For sets Z and Y in $A2$(Binary Image), erosion of Z by Y is denoted by $Z \ominus Y$

$$Z \ominus Y = \{a|(Y)a \subseteq Z\} \tag{3}$$

the collection of all points a such that Y, shifted or translated by a, is contained in A.

Opening In morphological image processing, the opening operation is a kind of image dilation that is followed by erosion. It is used to eliminate small, isolated white pixels (noise) while maintaining the general structure of larger white regions (objects). The opening operation is described as an erosion followed by a dilation, using the same structuring element for both operations. The erosion step reduces the white areas by removing isolated, tiny pixels, and the dilation step then enlarges the remaining white areas to their original size. The opening operation can help remove noise from binary images and in smoothing object boundaries in grayscale images.

The opening operation dilates the eroded image after first eroding it using the same structural element. The morphological opening can be used to eliminate fine details and thin lines from images while keeping the size and shape of the larger items.

$$Z \circ Y = (Z \ominus Y) \oplus Y \tag{4}$$

Closing The closure process uses the same structural element for both enlarging and contracting an image, respectively. Morphological closing can be used to fill in small gaps in the image while maintaining the size and shape of larger gaps and objects.

$$Z \bullet Y = (Z \oplus Y) \ominus Y \tag{5}$$

3.2 Feature extraction

A group of techniques known as feature extraction translate input data into fresh output features. Unsupervised learning is frequently used in feature extraction techniques to extract features. The amount of redundant data in the data collection is decreased with the aid of feature extraction. In the end, the data reduction speeds up the learning and generalization phases of the machine-learning process and enables the model to be built with less machine effort.

Features can be extracted from pre-processed images for analysis and classification. Texture features (Improved Binary Gabor Pattern, Scale Invariant Feature Transform, Gray-Level Co-occurrence Matrix), color features (Color Histogram), and shape features (Center of gravity, Orientation, Thickness, Anisotropy) can be used to capture patterns, structures, color distribution, and object shape and structure in the image.

3.2.1 Texture feature

Text feature refers to an attribute or characteristic of text data that can be used to describe, classify, or analyze it. Texture attributes such as CLBP, SIFT, and GLCM do not extract discernible patterns and formations that pertain to the texture of the foliage, a vital component in the process of identifying diseases.

Chi-squared Local Binary Pattern (CLBP) LBP is a well-liked texture analysis feature that is useful for several tasks, including object tracking, image classification, face recognition, etc. The LBP is the source of all other pattern-based features. It captures the intensity variations in each pixel's immediate surrounding area. In LBP, the distance between the pixels is computed randomly, and this affects the performance of the model. To resolve this issue, the chi-squared distance is computed. The CLBP is a variant of the Local Binary Patterns (LBP) algorithm that incorporates the Chi-squared distance metric. The advantages of using CLBP over traditional LBP include:

- Improved Discrimination Power: CLBP has improved discrimination power compared to traditional LBP, which means it can better differentiate between textures that are very similar in appearance. This is because the Chi-squared distance metric takes into account the differences in the intensity values of the pixels in the image, not just their binary patterns.

- Robustness to Noise: CLBP is more robust to noise in the input image compared to traditional LBP, which can result in more accurate texture representations even in the presence of noise.
- Better Histogram Comparison: CLBP provides a more meaningful comparison of histograms than traditional LBP, which can lead to improved performance in tasks such as image classification, texture analysis, and object recognition.
- Flexibility: CLBP can be used with different distance metrics, such as the Euclidean distance or the Manhattan distance, to improve its performance for a specific task or image type.
- Improved Computational Efficiency: CLBP can be computationally efficient compared to traditional LBP, especially when computing the LBP representation for large images.

To calculate the CLBP values using Eq. (6), a pixel's intensity value is compared with those of its neighbors within a radius R of the pixel.

Chi-square, denoted by the symbol X^2, is a method of analyzing data sets by comparing two pixels in the pre-processed image. The formula for the chi-squared test is frequently the sum of the squared discrepancies between the central and its surrounding pixel divided by the variance of the sample. It is mathematically denoted as Eq. (13),

$$X^2 = \sum (A_q - A_0)^2 / A_0 \tag{6}$$

where EV_i is represented as expected value and OV_i is represented as an observed value

$$LBP(a,b) = \sum_{q=1}^{8} 2^{q-1} * t_1\left(A_q - A_0\right) \tag{7}$$

where (a, b) represents the pixel location, $1 \le a \le M_1, 1 \le b \le M_2$ with $M_1 * M_2$ indicating the size of the image, A_q and A_0 denote the intensity value of the neighbor pixel and center pixel, respectively.

$$t_1(d) = \begin{cases} 1 & d \ge 0 \\ 0 & otherwise \end{cases} \tag{8}$$

Starting with $q = 1$ for the pixel to the right of the center pixel, the neighborhood pixel's index q increases from 1 to 8 in an anticlockwise direction. Figure 1 illustrates the calculation of LBP for a $3 * 3$ neighborhood. The histogram is created as in Eq. (8) and is used as a feature descriptor for the CLBP values.

$$Y(e) = \sum_{a=1}^{M_1} \sum_{b=1}^{M_2} t_2(LBP(a,b), e), e \in \left\{0, 1, \ldots., 2^{q-1}\right\} \tag{9}$$

Were,

$$t_2(d, y) = \begin{cases} 1 & d = y \\ 0 & otherwise \end{cases} \tag{10}$$

Scale Invariant Feature Transform (SIFT) Local features in images can be identified and characterized using the Scale-Invariant Feature Transform (SIFT) method. This feature is frequently applied in image processing. Differences in Gaussians (DoG) Space Generation, Key points Detection, and Feature Description are some of the SIFT operations.

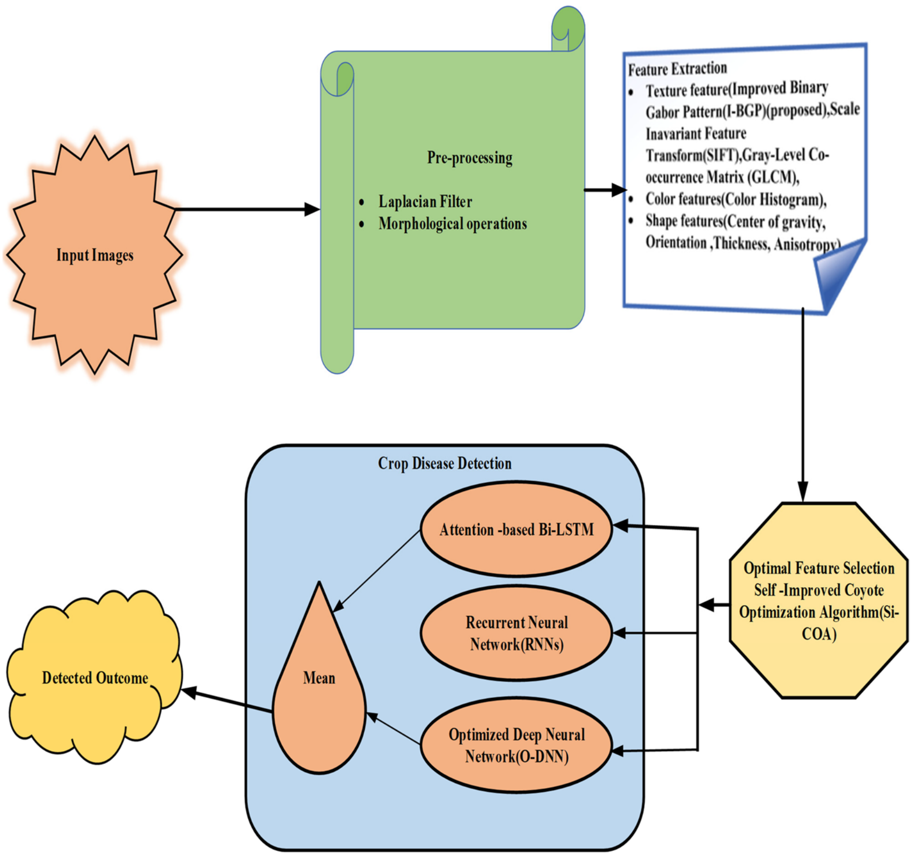

Input Images
Pre-processing
• Laplacian Filter
• Morphological operations
Feature Extraction
• Texture feature(Improved Binary Gabor Pattern(I-BGP)(proposed),Scale Inavariant Feature Transform(SIFT),Gray-Level Co-occurrence Matrix (GLCM),
• Color features(Color Histogram),
• Shape features(Center of gravity, Orientation ,Thickness, Anisotropy)
Optimal Feature Selection Self -Improved Coyote Optimization Algorithm(Si-COA)
Crop Disease Detection
Attention -based Bi-LSTM
Recurrent Neural Network(RNNs)
Optimized Deep Neural Network(O-DNN)
Mean
Detected Outcome

Fig. 1 Overall proposed model

SIFT features A popular feature detection method developed by Lowe [28] is SIFT. It derives robust feature points that are resistant to projective transformation, image rotation, scaling, limited affine distortion, and change in light. The following list summarises the primary steps involved in feature extraction using the SIFT algorithm:

Scale-space extrema detection: Using a SIFT detector, the feature points are found by looking for local maxima at various scales of the considered image using the difference of the Gaussian (DoG) function. Considering u (c, b) to be the input image, the following definitions can be used to specify the scale space:

$$\mathrm{O}(\mathrm{c},\mathrm{b},\sigma) = \mathrm{T}(\mathrm{c},\mathrm{b},\sigma) * \mathrm{u}(\mathrm{c},\mathrm{b}) \tag{11}$$

Gray-Level Co-occurrence Matrix (GLCM) The second-order statistical texture analysis method is called GLCM. To determine how frequently a particular combination of pixels is present in an image in a given direction and at a given distance, the spatial connection between pixels is examined. Each image is quantized into 16 grey levels (0–15) and 4 GLCMs (M) for each of the four angles of view (=0, 45, 90, and 135) to attain the desired d value of 1. Five characteristics (Eq. 12–15) are retrieved from each GLCM. Therefore, each image has 20 attributes. Before being passed to the classifiers, each feature is normalized to have a range of 0 to 1, and each classifier is given the same set of features.

$$\text{Energy} = \sqrt{\sum\nolimits_{r,q=0}^{T-1} (N(r,q))^2} \tag{12}$$

$$\text{Homogeneity} = \sum\nolimits_{r,q=0}^{T-1} \frac{N(r,q)}{1+(r+q)^2} \tag{13}$$

$$\text{Correlation} = \sum\nolimits_{r,q=0}^{T-1} \frac{N(r,q)}{1+(r+q)^2} \left[\frac{(r-q)(q-\overline{q})}{\sqrt{(\sigma_{r^2})(\sigma_{q^2})}} \right] \tag{14}$$

$$\text{Energy} = \sum\nolimits_{r,q=0}^{T-1} N(r,q)X(r,q)^2 \tag{15}$$

3.2.2 Color features

A property or attribute of an object or image that describes its hue, saturation, and brightness is referred to as its color. It is frequently used in image and video processing, computer vision, and graphics, where it is essential for locating and analyzing specific objects or regions in an image or video. Various color spaces, including RGB, HSV, HSL, and others, can be used to represent the color feature. Color characteristics are derived in order to capture variations in color that may suggest the presence of disease.

Color Histogram A color histogram is a visual depiction of the distribution of colors in an image used in image processing and photography. A color histogram for digital images shows the proportion of pixels that fall inside each of a predetermined set of color ranges that cover the whole color space of the image. The most popular method for identifying an image's color features is color histogram analysis. It depicts the scene from a different angle. It displays the color bins' frequency distribution in the image. Similar pixels are counted and stored. The global and local color histograms are two different forms of color histograms. An all-encompassing color descriptor that examines each statistical color frequency in an image is called a color histogram. Changes in translation, rotation, and angle of view difficulties are solved with it. Local color histograms concentrate on a specific area inside an image.

The SR is defined as indicated in Eq. (15) for a given pair of color histograms, C and B, with p bins for each.

$$SR(C,B) = \sum\nolimits_{r=1}^{p} min(C_r, B_r) \tag{16}$$

3.2.3 Shape features (Center of gravity, orientation, thickness, anisotropy)

Shape features are crucial because they offer a different way to describe an object than just employing its most salient properties, which helps to store fewer data. Shape features are extracted to capture the structural characteristics of the leaves, aiding in disease type differentiation.

Center of gravity The center of gravity is a point in an object where all of the mass of the object is concentrated. Fixed with shape Z is the area of the contour given as per Eq. (16).

$$Z = \frac{1}{2}\left|\sum_{r=0}^{M-1}(c_r b_{r+1} - c_{r+!} b_r)\right| \tag{17}$$

Orientation Object detection is employed in image processing and computer vision. In the confined area of an image, the technique counts instances of gradient orientation. This technique and Edge Orientation Histograms and Scale Invariant Feature Transformation are pretty comparable (SIFT).

$$p = \sqrt{y_1^2 + y_2^2 + y_3^2 \ldots \ldots \cdots + y_{36}^2} \tag{18}$$

$$u_{yr} = \left[\left(\frac{y_1}{p}\right), \left(\frac{y_2}{p}\right), \left(\frac{y_3}{p}\right) \ldots \ldots \left(\frac{y_{36}}{p}\right)\right] \tag{19}$$

Thickness The measure of how thick a thing is, as distinguished from the length or width of any of its surfaces

$$Thickness(k) = \left|p_2(k)\right| \tag{20}$$

Anisotropy Anisotropic diffusion, also known as Perona-Malik diffusion, is a technique used in image processing and computer vision that aims to reduce image noise without removing significant portions of the image content, usually edges, lines, or other details that are crucial for the interpretation of the image.

The diffusion in isotropic diffusion occurs along the gradient's direction since it uses a scalar-valued diffusivity. It may replace the scalar-valued diffusivity function with a diffusion tensor $W \in I\, 2\times 2$ to obtain an equation

$$f_g = div(W \cdot \nabla \sqcap) \tag{21}$$

where the matrix W depends on the edge-estimator ∇uσ and can be construed to fulfill desired properties.

3.3 Feature selection

Feature selection in crop disease detection refers to the process of identifying and choosing a subset of relevant and important features from a larger set of features to use in the classification or prediction of crop diseases. The objective is to choose the most illuminating and discriminative features that can raise the illness detection model's precision, efficacy, and interpretability. By removing traits that are redundant, irrelevant, noisy, or have poor predictive value, this is accomplished. This consists of the process of choosing the most suitable and enlightening attributes through the utilization of an optimization algorithm known as SI-COA, which aids in the reduction of the number of dimensions within the attribute space. Furthermore, it retains the most distinguishing attributes essential for the identification and diagnosis of diseases.

3.3.1 Optimal feature selection

While retaining the same level of true positive rates, optimal feature selection reduces mistakenly picked features by 50%. In this research work, the optimal features are elected using the new SI-COA model.

Self-Improved Coyote Optimization The COA is a brand-new metaheuristic optimization algorithm that, like a swarm and evolutionary algorithms, is characterized by a random population in its design. The species of Canis latrans serve as its inspiration. The suggested algorithm outlines how coyotes interact with one another in their social environment and how they adapt to it. The COA and grey wolf optimization (GWO) algorithms are completely different from one another. While the COA shows the coyotes' social structure and shared experiences, the GWO's algorithm describes the complete process of assaulting the prey. The Self-Improved Coyote Optimisation Algorithm (SI-COA) is employed to achieve optimal feature selection, hence mitigating the inclusion of erroneous features while preserving the genuine positive rates at a consistent level.

The drawbacks of using COA in optimal feature selection:

- Slow convergence: COYOTE can be slow to converge, especially for high-dimensional problems, due to its iterative nature.
- Local optima: COA may get stuck in local optima, which can lead to suboptimal solutions.
- Sensitivity to initial conditions: COYOTE is sensitive to the initial conditions, and the choice of the initial solution can affect the final results.
- Hyperparameter tuning: COYOTE requires tuning of several hyperparameters, such as the number of agents, the step size, and the number of iterations, which can be time-consuming and challenging.

To resolve these issues, and to enhance the detection accuracy of the proposed model, a new SI-COA model is introduced in this research work.

Step 1: Initialization

The first step of SI-COA is initializing the input parameters. Here the input parameters are the extracted features. Each group in this method has N_c coyotes and the population is divided into N_p groups. Coyote social behavior is the fitness function cost, and it is a potential solution. The following diagram illustrates how a set of design variables—the social behavior of the coyote c in a group p at a time t is represented. The cost of the fitness function is thought to be the coyote's adaption to its surroundings. The coyotes or agents are created at random throughout the search space during the COA's initialization procedure and are constructed as follows:

$$X_{c,j}^{p,t} = LB_j + r_j\left(UB_j - LB_j\right) \tag{22}$$

where LB the design variable j lower and upper limits are j and UB_j, respectively, and r_j is a random number between 0,1.

Random values between [0, 1] are used for variables such as social contact rates, group influence, and alpha weights. These values calculate the force of gravity and search agent advection. An arbitrary exploration technique is not effective in optimizing the hyperparameters of the DCNN model. Random numbers introduce variability and exploration, aiding in the search for optimal solutions in data mining.
Step 2: Fitness computation
Each coyote's fitness function is expressed as follows:

$$fit_c^{p,t} = \max(\text{Acc}) \tag{23}$$

Step 3- Group formation (proposed)
Coyotes initially form groups at random during the COA, yet occasionally they will switch to a different group. Here, Acc points to classification accuracy. The probability P_L which is expressed by the following equation, is connected with this coyote leave:

$$P_L = 0.0005.N_c^2 \tag{24}$$

Step 4- Eliminating disputes between search agents (proposed)
The suggested technique facilitates the collective exchange of coyote culture. During the collective exchange process, the disputes are eliminated based on the force of gravity, and social interaction among search agents. This collective exchange process is mathematically modelled as per Eq. (20). Here, Vector $\vec{E}$ is used to calculate the new search agent position to prevent conflicts between search agents (i.e., other tunicates).

$$\begin{aligned} \vec{E} &= \frac{\vec{S}}{\vec{D}} \\ \vec{S} &= v_2 + v_3 - \vec{P} \\ \vec{P} &= 2 \cdot v_1 \\ \vec{D} &= X_{Min} + v_1 \cdot X_{Max} - X_{Min} \end{aligned} \tag{25}$$

Therefore, $\vec{E}$ represents the force of gravity, while $\vec{P}$ depicting the advection of the search agent. The range [0, 1] is where the random numbers for the variables $v_1, v_2\ v_3$ are located. $\vec{D}$ stands for the forces of social interaction among search agents.
Where X_{Min} and X_{Max} stand for the primary and secondary rates of social contact. The values of X_{Min} and X_{Max} are 1 and 4. The architecture of the Tunicate Swarm Algorithm is shown in Fig. 1.
Step 5- Environmental adaptation
The coyote thought to have the best environmental adaptation is the alpha coyote, which can be calculated analytically as follows:

$$alpha^{p,t} = 1X_c^{p,t} for \ \min fit_c^{p,t} \tag{26}$$

Step 6- Position updating
The suggested method encourages group exchange of coyote culture. The alpha coyote, which may be determined analytically as follows, is regarded to have the finest coping mechanisms with its surroundings:

$$cul_j^{i,t} = \left\{ \begin{array}{cc} O^{p,t}\frac{N_c+1}{2},j & N_C \ is \ odd \\ \frac{O^{p,t}\frac{N_c}{2},j+O^{p,t}\left(\frac{N_c+1}{2}\right),j}{2}, & otherwise \end{array} \right\} \tag{27}$$

where $O^{p,t}$ represents the coyotes in group p rated social conditions at time t for design variable j. The COA takes into account the coyote's life cycle, including its birth and death. Coyotes are born as a result of a combination of the social behavior of two randomly chosen parents within the search area and an environmental element. Following is how this life event is expressed:

$$pup_j^{p,t} = \left\{ \begin{array}{c} X_{r_{1,j}}^{p,t}, r \ and \ j < P_s orj = j_1 \\ X_{r_{2,j}}^{p,t}, r \ and \ j \geq P_s + P_a orj = j_2 \end{array} \right\} \tag{28}$$

Were,

$$P_s = {}^1/_D$$
$$P_a = \frac{(1-P_s)}{2}$$

where j_1 and j_2 are two random design variables, P_s and p an is the scatter and association probabilities, R_j stands for a random number inside the design variable limit, and r_j is a random number that is between 0 and 1. where $r_1 andr_2$ are random coyotes within the group p. ω is the number of coyotes in the group and stands for the coyotes with the worst solutions. Using the components δ_1 and δ_2, the COA illustrates how culture interacts within groups. δ_1 represents the cultural difference between a random coyote c_{r_1} and the group's alpha coyote, and δ_2 indicates the cultural difference between a random coyote c_{r_2} and the group's cultural tendencies.

$$\delta_1 = alpha^{p,t} - X_{c_{r_1}}^{p,t} \tag{29}$$

$$\delta_2 = cul^{p,t} - X_{c_{r_2}}^{p,t} \tag{30}$$

The alpha coyote and group influence then update the coyote's social behavior as follows:

$$n.X_C^{p,t} = X_c^{p,t} + r_1\delta_1 + r_2\delta_2 \tag{31}$$

where r1 and r2 are random values between [0, 1] that express the group influence and alpha weights, respectively. The following criteria are used to assess the coyotes' new fitness value:

$$nfit_c^{p,t} = f\left(nX_c^{p,t}\right) \tag{32}$$

It is replaced by the new social behavior if it is superior to the old one, as in:

$$X_c^{p,t+1} = \left\{ \begin{array}{c} n.X_c^{p,t}, nfit_c^{p,t} < fit_c^{p,t} \\ X_C^{p,t}, \ otherwise \end{array} \right\} \tag{33}$$

The social conduct that best adapts to the environment is selected as the optimal option at the end of this procedure. The $X_c^{p,t+1}$ is the best solution acquired so far, and it points to the position of the optimal features.

3.4 Crop disease detection

The detection of plant pests and illnesses is an important field for machine vision research. Using machine vision technology is a method for examining photographs of plants to see if they are infested with pests or diseases. The proposed approach utilises Attention-based Bi-LSTM, RNNs, and O-DNN to identify crop diseases. The optimal adjustment of the weighting of the DNN is performed using the SI-COA. The performance of the algorithm is assessed by employing a range of performance metrics, including Precision, Recall, F1-Score, Accuracy, Sensitivity, Specificity, Negative Predictive Value (NPV), False Negative Rate (FNR), False Positive Rate (FPR), and Matthews Correlation Coefficient (MCC).

3.4.1 Bidirectional LSTM algorithm

BI-LSTM network using both sequences of past and future input data, bi-LSTM is developed. Two linked layers are used to process the input data. Bi-directional LSTM predicts or tags the sequence of each element using a finite sequence based on the context of items from the past and future. The outcome of two LSTMs is this Bi-LSTM is developed to train a network using input data sequences from both the past and the future. Utilizing two linked layers, the input data are processed. The bi-directional LSTM uses the context of elements in the past and future to anticipate or tag the sequence of each element using a constrained sequence. Two LSTMs resulted in this.

$$a_h^t = \sum\nolimits_{l=1}^{l} x_1^t w_{lh} + \sum\nolimits_{h't>0}^{H} b_{h'}^{t-1} wh'h \tag{34}$$

$$a_h^t = \theta_h\left(a_n^t\right) \tag{35}$$

Two distinct LSTM networks make up the bidirectional LSTM, one of which processes the sequence from beginning to end and the other from end to end. As an input to the following layer or for prediction, the outputs from these two LSTMs are then combined. In several NLP applications, including sentiment analysis, machine translation, and text classification, bidirectional LSTMs have been effectively used. These tasks can benefit from a better performance by capturing contextual information from both sides. Show the given below Fig. 2.

The Bi-LSTM is an extension of the RNN architecture that is able to digesting incoming data sequences by considering both past and future knowledge. The model is made up of two interrelated sections that successfully handle the input data in a bilateral manner. This design enables the model to precisely represent the relationships that are that characterise both the following and following context of each element within the sequence. The Bi-LSTM model is adept at forecasting or allocating labels to each element in a sequence using the historical context from both previous and succeeding items. The earlier objective is achieved by using two sets of LSTM models, the first LSTM processing the sequence in the direction of forward movement and the additional LSTM handling the pattern in the reverse style. The ultimate forecast or labelling for each of the components in the sequence can be determined via the combination of the outputs generated by the two LSTM models. Bi-LSTM models are used in

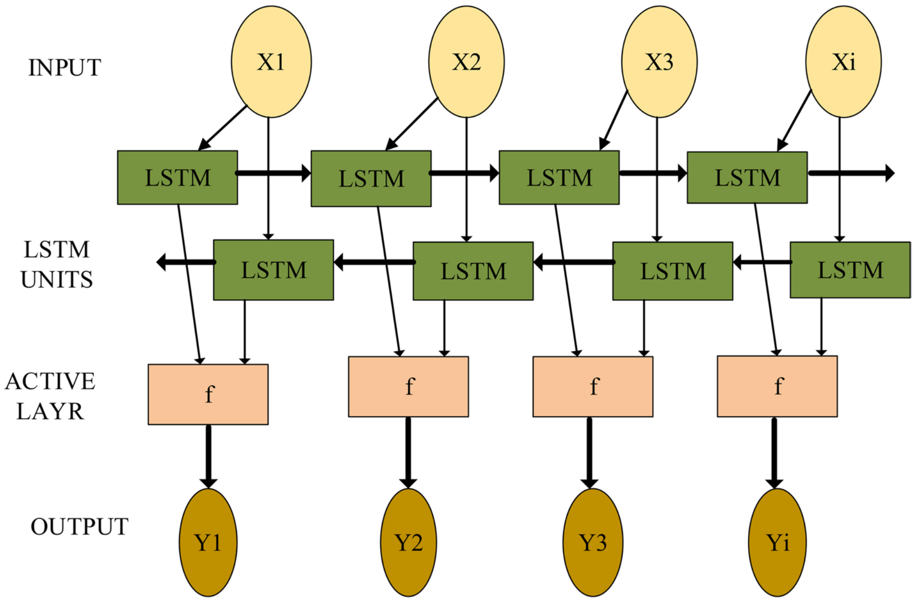


Fig. 2 Flowchart—bidirectional LSTM algorithm

many applications, like crop disease detection, demonstrating increased performance in compared to other models.

3.4.2 Recurrent neural networks

Recurrent neural networks (RNNs) were created specifically to get over feedforward neural networks' constraints when modeling sequences. RNN networks are made up of components, or neurons, that may remember prior observations. This enables RNNs to model the input data using both recent and old information witnessed comments. The Long Short-Term Memory (LSTM) architecture, one of the suggested RNNs, performed very well at modeling the long-range temporal dependencies of the input data. Formally, the LSTM unit recursively carries out the following operations given an input vector $x(t)$ and the current observation (let's say $x(t+1)$) :

$$i(t) = \sigma\left(W_C\left[x(t), h(t-1) + b_i\right]\right) \tag{36}$$

$$\overline{C}(t) = \tanh\left(W_c[x(t), h(t-1)] + b_c\right) \tag{37}$$

$$f(t) = \sigma(W_f\left[x(t)[x(t), h(t-1)] + b_f\right] \tag{38}$$

$$C(t) = f(t) \odot \mathrm{c}(t-1) + i(t) \odot \overline{C}(t) \tag{39}$$

$$o(t) = \sigma\left(W_o[x(t), h(t-1)] + b_o\right) \tag{40}$$

$$h(t) = ot \odot \tanh(C(t)) \tag{41}$$

where $\bigodot$ is the matrix multiplication done element by element. While $\tilde{i}\tilde{c}fco$ are referred to as input, input modulation, forget, cell, and output gates and jointly conduct operations to make the LSTM able to select the information to remember and to forget from the input data, W_i, W_c, W_f, W_o and b_i, b_c, b_f, b_o learnable kernels and biases, respectively. When combined with x, the hidden state h encodes the information that the LSTM unit has learned from prior observations.

3.4.3 Optimized deep neural network

The process of minimization (or maximization) of any mathematical statement is called optimization. Programs or methods known as optimizers change the weights and learning rate of the neural network to reduce losses. Optimizers can resolve optimization problems by decreasing the function. A deep network is a hierarchical composition of nonlinear functions σ and linear functions of an input datum, , $\in, R^d$, at each level. A neural network's output, denoted by the symbol $y(x;\xi)$, can be expressed as:

$$y(x;\xi) = \sigma'\left(x^P\sigma\left(x^{P-2}\ldots.\sigma\left(x'\xi\right)\right)\ldots\ldots..\right) \tag{42}$$

where $x^1, \ldots\ldots\ldots x^{p-1} \in R^{d*d}$ and $x^p \in R^{d*k}$. After each of them has been rearranged as a vector, we'll use the notation $x \in R^n$ to represent their concatenation. The arguments of the functions $\sigma(.)$ are applied point-wise to each component. In a network using rectified linear units, for instance, we have $\sigma(z) = \max(0, z) forz \in R^d$. The last nonlinear set is $\sigma'(z) = 1_{\{z\geq 0\}}$ Consequently, a neural network is a function with n parameters that generates an output $y(x;\xi) \in \{1, \ldots\ldots\ldots k\}$ in every inputξ.

Given a limited sample of inputs and outputs, the challenge of "training" or "supervised learning" a deep network for image categorization $\mathrm{D} = \{\xi^1, y'\}_{i=1}^{N}$. One wants to reduce empirical loss.

$$f(x) := \frac{1}{N}\sum_{i=1}^{N} f_i(x); \tag{43}$$

where $f_i(x)$ is a loss function that calculates the difference between the forecasts $y\left(x;\xi^i\right)$ in samples ξ^i the actual label y^i is the zero–one loss, for instance $f_i(x) := 1_{\{y(x;\ \xi^i)\neq y'}$

whereas a quadratic penalty could be imposed for a loss in extrapolation $f_i(x) := \frac{1}{2}||y\left(x;\ \xi^i\right) - y^i||_2^2$

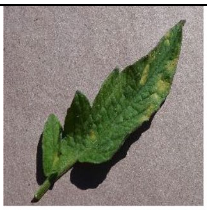
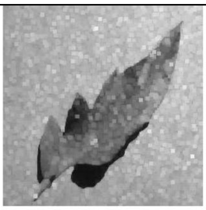
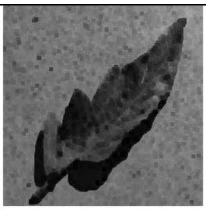
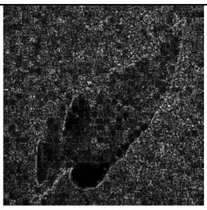
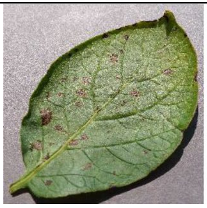
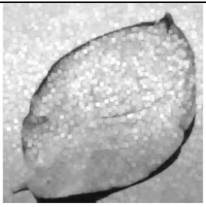
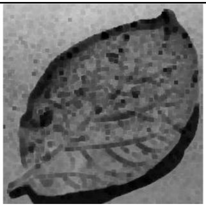
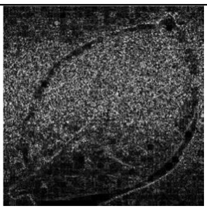
We will ignore the details of the loss function in the following section and look at training a deep network as a general non-convex optimization problem given by

$$x^* = arg_x \min f(x) \tag{44}$$

Deep network classification challenges were the focus of the development and testing of the specific relaxation and regularisation strategies that we discuss.

Mean The mean value of a set of numerical data is calculated using the mean, a measure of central tendency, to provide a summary. It is calculated by adding up all the values in a data collection and dividing the total by the number of values. The mean is also commonly referred to as the average. It is a commonly used statistical measure that gives a rough

Table 2 Pre-processed findings

Input	Output		
	Dilation	Erosion	Laplacian
			
Class 1	(a)	(b)	(c)
			
Class 2	(d)	(e)	(f)

idea of the typical value in a data set and is especially useful when the data is normally distributed.

The mean value is detected by the outcomes from Bi-LSTM and QDNN. It utilises the formula as shown in Eq. (41).

$$Mean = \frac{Sum\ of\ all\ outcomes}{Total\ number\ of\ outcomes} \tag{45}$$

4 Result and discussion

The suggested methodology is put into practice using Python programming. To evaluate the performances, performance matrices such as specificity, accuracy, sensitivity, MCC, NPV, FPR, FNR, f-measure, and precision are employed. The proposed method is compared to the existing techniques such as COA, RSO, SVM and CNN.

Pre-processing of raw images utilizing classifications from the dataset containing healthy and diseased plant leaves images are categorized as Class 1 and Class 2 type and they are processed with Dilation filter, Erosion filter, and Laplacian filter and its outcome images are tabulated in Table 2 (Figure (a) to Figure (f)).

Images from the proposed study that accurately classify plant leaves with disease and healthier leaves with labelling for ablation test as depicted in Fig. 3.

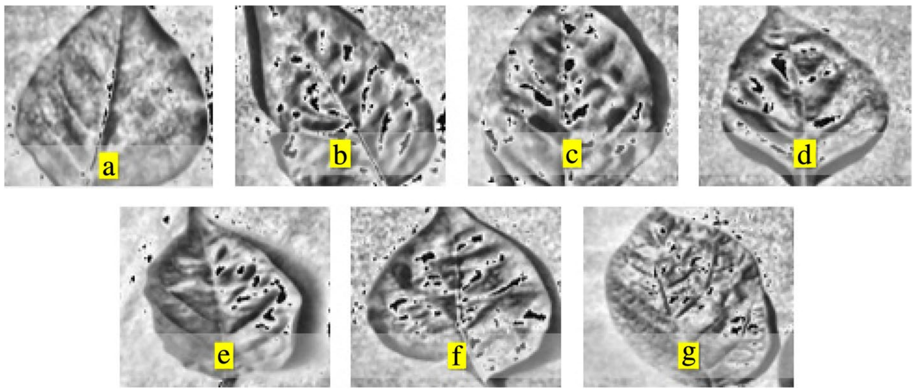


Fig. 3 Accurate disease detected images of pepper plant leaf for ablation study

The effectiveness of the proposed model strategy is analysed by conducting ablation study which illustrated in Fig. 3(a)-(g) respectively for (a) without optimization,(b) with only Attention-based Bi-LSTM,(c) with only RNNs,(d) with only DNN,(e) with Attention-based Bi-LSTM and RNN,(f) with Attention-based Bi-LSTM,RNN and DNN,(g)with voting-based ensembled-deep-learning paradigm (Attention-based Bi-LSTM, RNN, and O-DNN (with SI-COA optimization)).

4.1 Overall performance of the proposed model-testing metrices-learn rate—0

The performance of the proposed model was evaluated against four other models: COA, RSO, SVM, and CNN. The evaluation was done using various performance metrics during testing, with a learning rate of 0. The proposed model showed the best performance among all the models, with an accuracy of 0.921312. This value was higher than the accuracies of COA, RSO, SVM, and CNN, which were 0.850394, 0.840053, 0.792073, and 0.859716, respectively. Precision, which measures the percentage of true positive predictions among all positive predictions, was also the highest for the proposed model, with a value of 0.913729. This value was higher than that of COA, RSO, SVM, and CNN, which were 0.843404, 0.833138, 0.758713, and 0.902190, respectively. Sensitivity, which measures the percentage of true positive predictions among all actual positive samples, was highest for

Table 3 Performance metrics of the proposed model -testing metrics learn rate -0

	COA	RSO	SVM	CNN	Proposed
Accuracy	0.850394	0.840053	0.792073	0.859716	**0.921312**
Precision	0.843404	0.833138	0.758713	0.902190	0.913729
Sensitivity	0.857555	0.847117	0.832548	0.814393	0.929060
Specificity	0.843328	0.833082	0.752135	0.904439	0.913668
F-Measure	0.823907	0.813878	0.768261	0.830885	0.892606
MCC	0.887904	0.883248	0.797378	0.851411	0.968686
NPV	0.857525	0.847107	0.832032	0.825198	0.929049
FPR	0.025971	0.020804	0.023765	0.020748	0.020199
FNR	0.009870	0.003646	0.006366	0.004296	0.003540

Table 4 Overall performance metrics of the proposed model -testing metrics-learn rate–1

	COA	RSO	SVM	CNN	Proposed
Accuracy	0.847498	0.893477	0.881103	0.878181	**0.912142**
Precision	0.840532	0.886122	0.843993	0.921567	0.904634
Sensitivity	0.854635	0.900990	0.926126	0.831884	0.919813
Specificity	0.840456	0.886063	0.836676	0.923864	0.904574
F-Measure	0.821101	0.865638	0.854614	0.848730	0.883722
MCC	0.884881	0.939419	0.887004	0.869697	0.959045
NPV	0.854605	0.900980	0.925553	0.842921	0.919802
FPR	0.025882	0.022778	0.026436	0.021194	0.019998
FNR	0.009836	0.003992	0.007082	0.004388	0.003505

the proposed model, with a value of 0.929060. The values of COA, RSO, SVM, and CNN for this metric were 0.857555, 0.847117, 0.832548, and 0.814393, respectively. The proposed model also showed the highest values for NPV and specificity, indicating its ability to correctly identify true negative samples. The values of NPV for the proposed model, COA, RSO, SVM, and CNN were 0.929049, 0.857525, 0.847107, 0.832032, and 0.825198, respectively. Finally, the proposed model had the lowest values for FPR and FNR, which measure the percentage of false positive and false negative predictions, respectively. This indicates that the proposed model had better performance in correctly classifying true negatives and true positives, respectively. Overall, the proposed model demonstrated superior

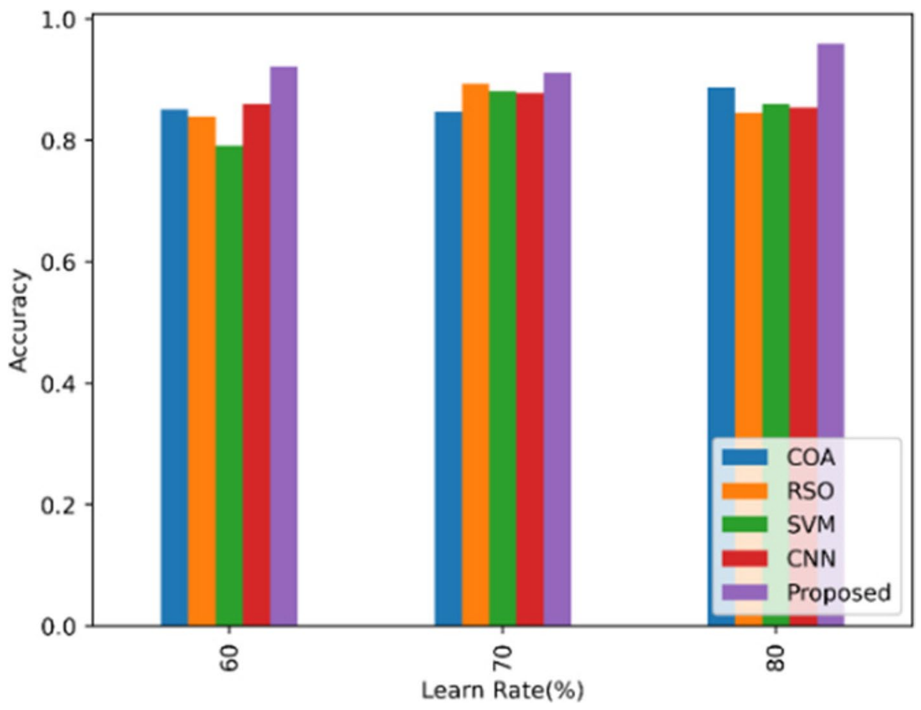


Fig. 4 Accuracy of the proposed model

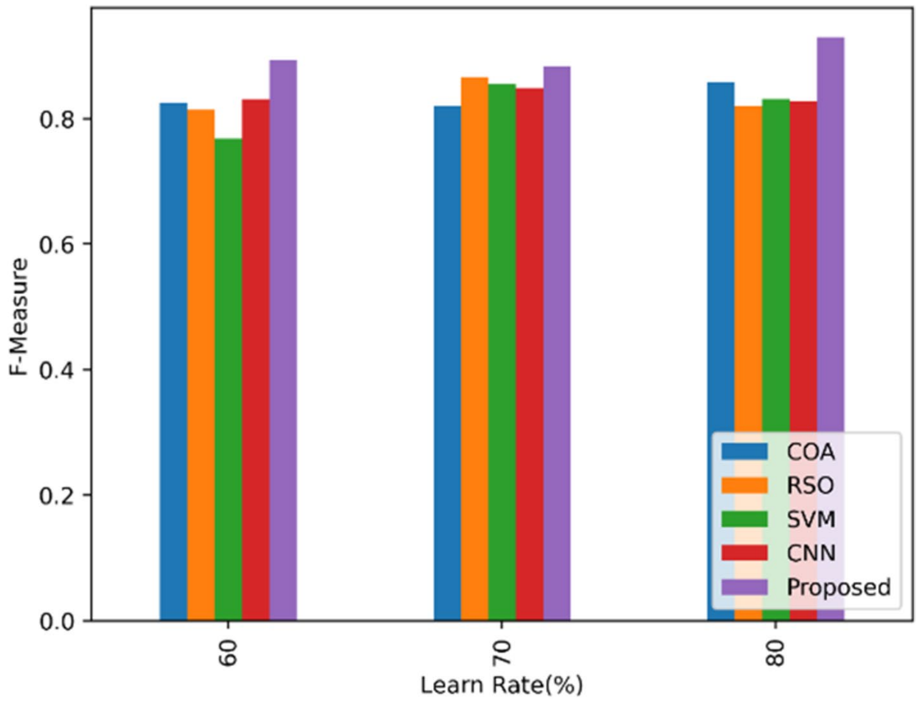


Fig. 5 F-measure of the proposed model

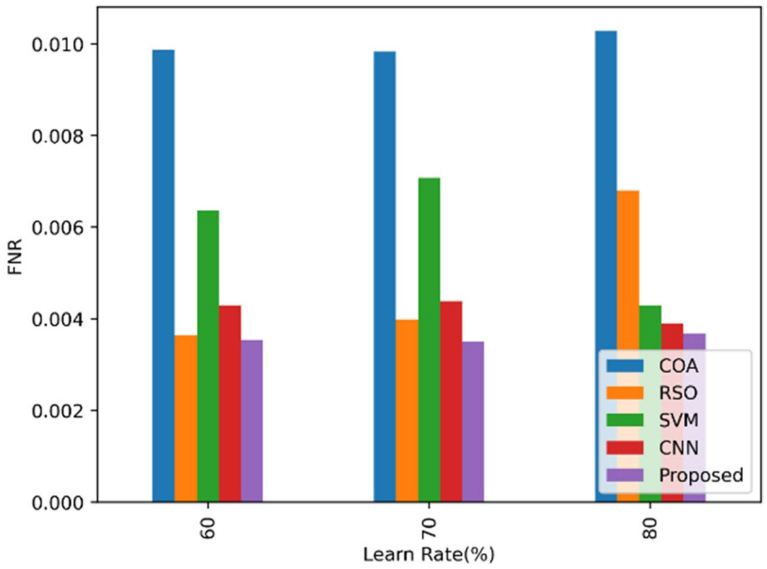


Fig. 6 FNR of the proposed model

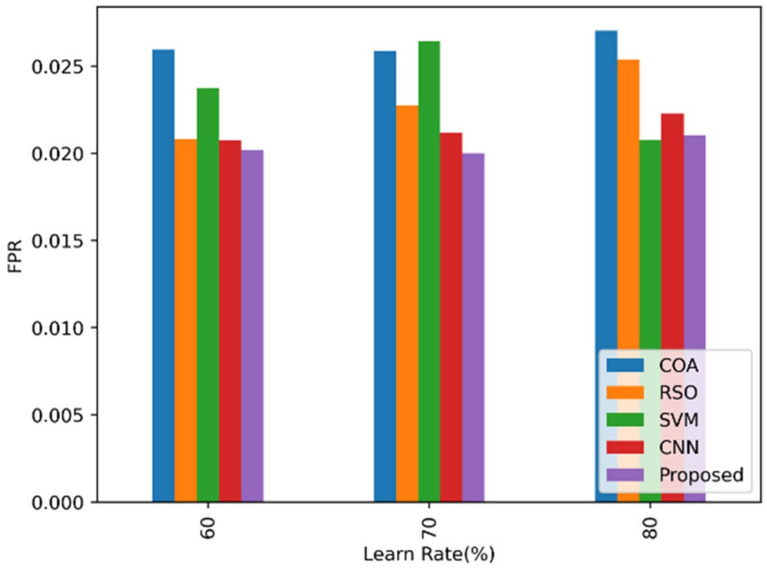


Fig. 7 FPR of the proposed model

performance in accurately classifying the data with high precision and sensitivity, making it a promising approach for the task at hand.

Table 3 shows the overall performance metrics of the proposed model with a learning rate of -1 on the testing data. The proposed model achieved an accuracy of 0.912142, outperforming the COA, RSO, SVM, and CNN models in terms of accuracy. The precision

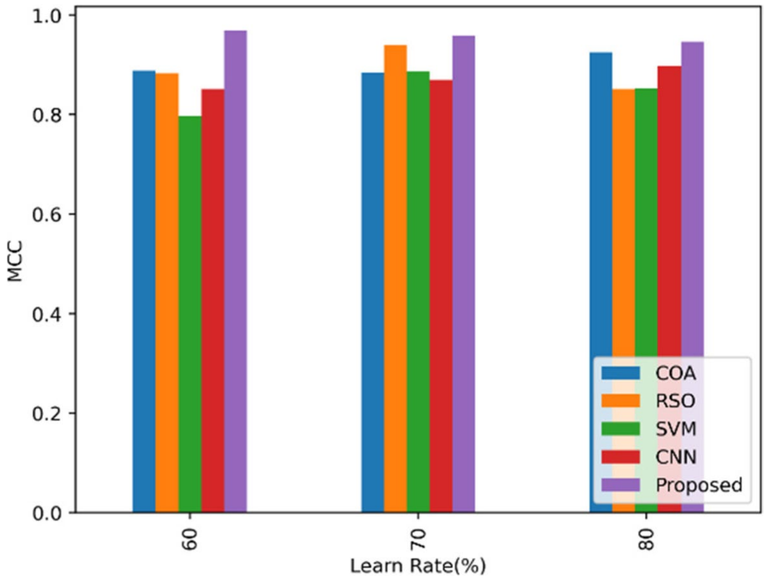


Fig. 8 MCC of the proposed model

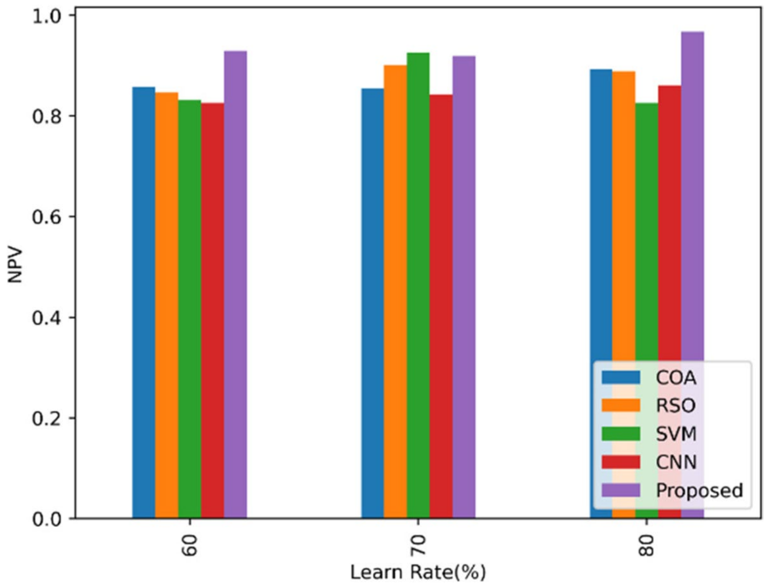


Fig. 9 NPV of the proposed model

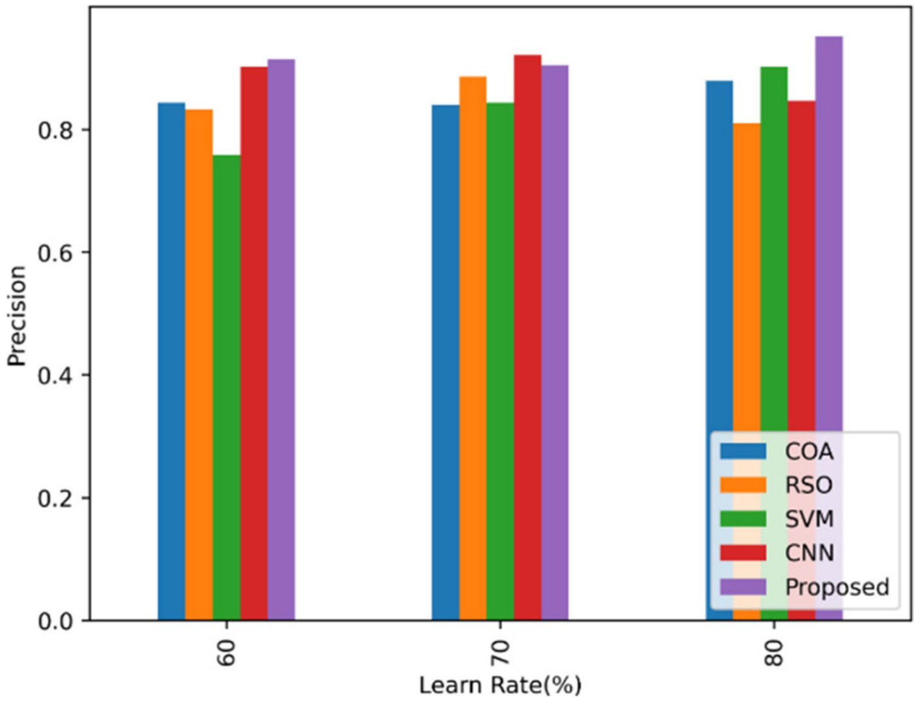


Fig. 10 Precision of the proposed model

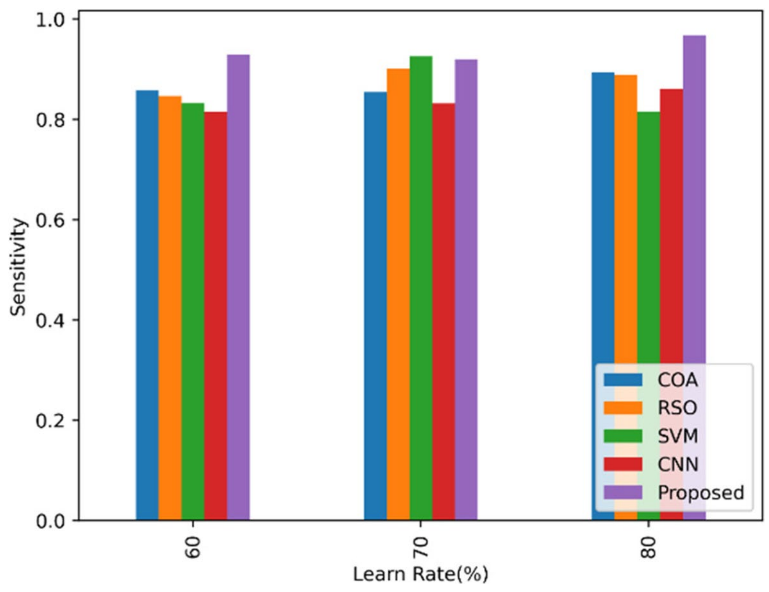


Fig. 11 Sensitivity of the proposed model

of the proposed model is 0.904634, which is higher than the precision of the COA, RSO, SVM, and CNN models. The proposed model also achieved a higher sensitivity of 0.919813 compared to the other models. However, the specificity of the proposed model

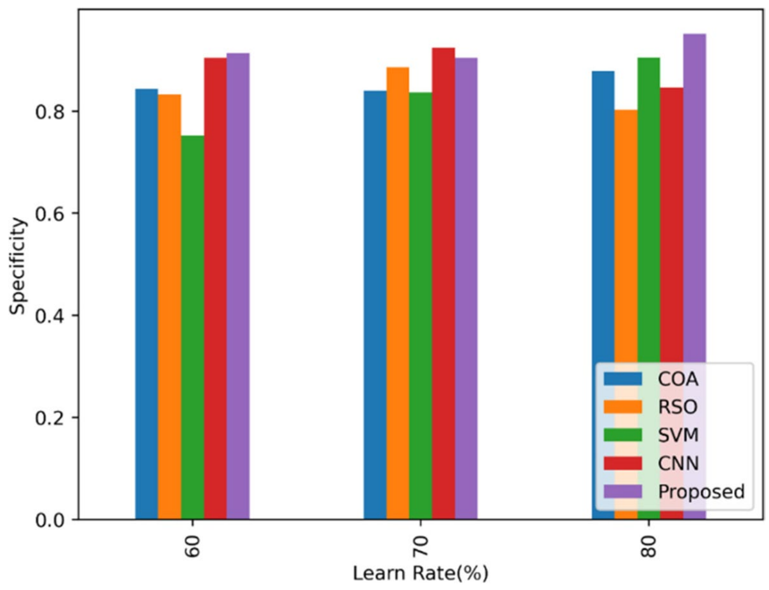


Fig. 12 Specificity metrics of the proposed model

Table 5 Overall performance metrics of the proposed model -testing metrics-learn rate–2

	COA	RSO	SVM	CNN	Proposed
Accuracy	0.886017	0.845859	0.860259	0.853922	**0.959907**
Precision	0.878734	0.810233	0.902759	0.846893	0.952005
Sensitivity	0.893478	0.889081	0.814907	0.861103	0.967978
Specificity	0.878655	0.803209	0.905010	0.846836	0.951942
F-Measure	0.858420	0.820429	0.831409	0.827315	0.929998
MCC	0.925099	0.851523	0.851948	0.897830	0.946515
NPV	0.893447	0.888531	0.825719	0.861093	0.967967
FPR	0.027059	0.025379	0.020761	0.022288	0.021046
FNR	0.010283	0.006799	0.004299	0.003906	0.003689

is slightly lower than that of the SVM and CNN models. The F-measure of the proposed model is 0.883722, which is higher than the F-measure of the COA and RSO models but lower than that of the SVM and CNN models. The MCC of the proposed model is 0.959045, indicating a strong correlation between the predicted and actual labels. The NPV of the proposed model is 0.919802, which is higher than that of the COA, RSO, SVM, and CNN models. The false positive rate (FPR) of the proposed model is 0.019998, indicating a low rate of false positives. Overall, the proposed model performs well in terms of accuracy, precision, sensitivity, F-measure, MCC, NPV, and FPR.

Table 4 presents the overall performance metrics of the proposed model with testing metrics learn rate -2. The accuracy of the proposed model is 0.959907, which is the highest among all the models evaluated in this study. The precision of the proposed model is 0.952005, while the sensitivity is 0.967978, both of which are also the highest among all the models. The specificity of the proposed model is 0.951942, which is slightly lower than some of the other models. The F-measure of the proposed model is 0.929998, while the Matthews correlation coefficient (MCC) is 0.946515. The negative predictive value (NPV) of the proposed model is 0.967967, which is the highest among all the models. The false positive rate (FPR) of the proposed model is 0.021046, which is also lower than some of the other models. The false negative rate (FNR) of the proposed model is 0.003689, which is the lowest among all the models. These results suggest that the proposed model has the best overall performance in terms of accuracy, precision, sensitivity, NPV, and FNR. Show the given below Figs. 4, 5, 6, 7, 8, 9, 10, 11, and 12, Table 5.

The suggested crop disease detection model outperformed other models in every performance measure when compared to other models, including COA, RSO, SVM, and CNN.

5 Conclusion

In this research, a new voting-based ensembled-deep-learning paradigm was proposed for crop disease detection, which included four major phases: pre-processing, segmentation, feature extraction, optimal feature selection, and crop disease detection. The proposed model was designed to improve the accuracy and efficiency of crop disease detection, which is a critical task for ensuring food security. The pre-processing phase involved the application of a Laplacian Filter and morphological operations to the collected raw images. Then ROI identification using K-means Clustering is processed. From the ROI regions, texture, color, and shape features were extracted using techniques such as Chi-squared

Local Binary Pattern (CLBP), Scale Invariant Feature Transform (SIFT), and Gray-Level Co-occurrence Matrix (GLCM). The optimal features were selected using a Self-Improved Coyote Optimization Algorithm (SI-COA) by mitigating the inclusion of erroneous features while preserving the genuine positive rates at a consistent level. And a voting-based ensembled-deep-learning paradigm was used for crop disease detection, which included Attention-based Bi-LSTM, Recurrent Neural Networks (RNNs), and an Optimized Deep Neural Network (O-DNN) where the weight of DNN was optimized by SI-COA algorithm. The performance of the proposed model was evaluated using various metrics such as Precision, Recall, F1-Score, Accuracy, Sensitivity, Specificity, NPV, FNR, FPR, and MCC with state-of-the-art models and it proven its improved effectiveness specially in overall accuracy values like 0.921312, 0.912142, 0.959907 respectively for learn rates 0,1,2 because of the mean value-based hybrid model strategy operation. In the future, this proposed model can be further improved by incorporating new deep-learning techniques and advanced feature extraction methods. Additionally, the model can be applied to large-scale datasets to evaluate its scalability and effectiveness in real-world scenarios. Furthermore, the proposed model can be extended to other domains such as plant disease detection, pest detection, and weed detection, which can contribute to sustainable agriculture practices.

Funding No fund received for this project.

Data availability All the data is collected from the simulation reports of the software and tools used by the authors. Authors are working on implementing the same using real world data with appropriate permissions.

Declarations

Conflicts of interest The authors declare that they have no conflict of interest.

References

1. Kumar R, Kumar R, Prakash O (2019) The impact of chemical fertilizers on our environment and ecosystem Chapter-5 the impact of chemical fertilizers on our environment and ecosystem. 2019
2. Kumar R, Kumar R, Prakash O (2019) Chapter-5 the impact of chemical fertilizers on our environment and ecosystem. Chief Ed 35:69
3. Chowdhury ME, Rahman T, Khandakar A, Ayari MA, Khan AU, Khan MS, Al-Emadi N, Reaz MBI, Islam MT, Ali SHM (2021) Automatic and reliable leaf disease detection using deep learning techniques. AgriEngineering 3(2):294–312
4. Tian M, Wei S, Bian R, Luo J, Khan HA, Tai H, Kondo H, Hadidi A, Andika IB, Sun L (2022) Natural cross-kingdom spread of apple scar skin viroid from apple trees to fungi. Cells 11(22):3686
5. Chandra M, Redkar S, Roy S, Patil P (2020) Classification of various plant diseases using deep siamese network. https://www.researchgate.net/profile/Manish-Chandra-3/publication/341322315_CLASSIFICATION_OF_VARIOUS_PLANT_DISEASES_USING_DEEP_SIAMESE_NETWORK/links/5ebaa82f299bf1c09ab52e48/CLASSIFICATION-OF-VARIOUS-PLANT-DISEASES-USING-DEEP-SIAMESE-NETWORK.pdf
6. Liu B, Ding Z, Tian L, He D, Li S, Wang H (2020) Grape leaf disease identification using improved deep convolutional neural networks. Front Plant Sci 11:1082
7. Mandi K, Patnaik NM (2019) Mobile apps in agriculture and allied sector: an extended arm for farmers. Agric Updat 14(4):334–342
8. Ashwinkumar S, Rajagopal S, Manimaran V, Jegajothi B (2022) Automated plant leaf disease detection and classification using optimal MobileNet based convolutional neural networks. Mater Today: Proc 51:480–487
9. Ji M, Zhang L, Wu Q (2020) Automatic grape leaf diseases identification via UnitedModel based on multiple convolutional neural networks. Inf Process Agric 7(3):418–426

10. Sowmya BJ, Shetty C, Seema S, Srinivasa KG (2020) Utility system for premature plant disease detection using machine learning. In: Hybrid Computational Intelligence. Academic Press, pp 149–172
11. Roy AM, Bose R, Bhaduri J (2022) A fast accurate fine-grain object detection model based on YOLOv4 deep neural network. Neural Comput Appl 34(5):3895–3921
12. Roy AM, Bhaduri J (2021) A deep learning enabled multi-class plant disease detection model based on computer vision. Ai 2(3):413–428
13. Haque IRI, Neubert J (2020) Deep learning approaches to biomedical image segmentation. Inform Med Unlocked 18:100297
14. Kabir MM, Ohi AQ, Mridha MF (2021) A multi-plant disease diagnosis method using convolutional neural network. Comput Vis Mach Learn Agric 99–111
15. Chen X, Wu S, Shi C, Huang Y, Yang Y, Ke R, Zhao J (2020) Sensing data supported traffic flow prediction via denoising schemes and ANN: a comparison. IEEE Sens J 20(23):14317–14328
16. Osorio K, Puerto A, Pedraza C, Jamaica D, Rodríguez L (2020) A deep learning approach for weed detection in lettuce crops using multispectral images. AgriEngineering 2(3):471–488
17. Ferentinos KP (2018) Deep learning models for plant disease detection and diagnosis. Comput Electron Agric 145:311–318
18. Barbedo JGA (2019) Plant disease identification from individual lesions and spots using deep learning. Biosys Eng 180:96–107
19. Guo Y, Zhang J, Yin C, Hu X, Zou Y, Xue Z, Wang W (2020) Plant disease identification based on deep learning algorithm in smart farming. Discret Dyn Nat Soc 2020:1–11
20. Abbas A, Jain S, Gour M, Vankudothu S (2021) Tomato plant disease detection using transfer learning with C-GAN synthetic images. Comput Electron Agric 187:106279
21. Zhao Y, Chen Z, Gao X, Song W, Xiong Q, Hu J, Zhang Z (2021) Plant disease detection using generated leaves based on DoubleGAN. IEEE/ACM Trans Comput Biol Bioinf 19(3):1817–1826
22. Hassan SM, Jasinski M, Leonowicz Z, Jasinska E, Maji AK (2021) Plant disease identification using shallow convolutional neural network. Agronomy 11(12):2388
23. Chen J, Zhang D, Zeb A, Nanehkaran YA (2021) Identification of rice plant diseases using lightweight attention networks. Expert Syst Appl 169:114514
24. Pandian JA, Kumar VD, Geman O, Hnatiuc M, Arif M, Kanchanadevi K (2022) Plant disease detection using deep convolutional neural network. Appl Sci 12(14):6982
25. Hassan SM, Maji AK (2022) Plant disease identification using a novel convolutional neural network. IEEE Access 10:5390–5401
26. Pandian JA, Kanchanadevi K, Kumar VD, Jasińska E, Goňo R, Leonowicz Z, Jasiński M (2022) A five convolutional layer deep convolutional neural network for plant leaf disease detection. Electronics 11(8):1266
27. Geetharamani G, Pandian A (2019) Identification of plant leaf diseases using a nine-layer deep convolutional neural network. Comput Electr Eng 76:323–338
28. Singh SP, Bhatnagar G (2020) Chapter 1 - perceptual hashing-based novel security framework for medical images. In: Singh AK, Elhoseny M (eds) Intelligent data security solutions for e-Health applications. Academic Press, pp 1–20. https://doi.org/10.1016/B978-0-12-819511-6.00001-7