SADCNN-ORBM: a hybrid deep learning model based citrus disease detection and classification

Ashok Kumar Saini[1], Roheet Bhatnagar[2], Devesh Kumar Srivastava[3]
[1,2]Department of Computer Science and Engineering, Faculty of Engineering, Manipal University Jaipur, Jaipur, India
[3]Department of Information Technology, Faculty of Engineering, Manipal University Jaipur, Jaipur, India

Article Info

Article history:

Received Sep 8, 2023
Revised Nov 19, 2023
Accepted Dec 5, 2023

Keywords:

Artificial intelligence
Citrus disease
Deep learning
Dilation
Machine learning
Texture feature

ABSTRACT

Citrus disease has a significant influence on agricultural productivity these days, so technology based on artificial intelligence has been developed for creating computer vision (CV) models. By spotting disease in its early stages and enabling necessary productivity actions, CV in agriculture improves the production of agricultural goods. In this paper, we developed a CV-based citrus disease detection model called the self-attention dilated convolutional neural network−optimized restricted Boltzmann machine (SADCNN-ORBM) model, which consists of two crucial parts: a SADCNN for disease segmentation and an ORBM optimized by the self-adaptive coati optimization (SACO) algorithm to improve the classification performance of diseases, which successfully divides the disease type into three groups: anthracnose, melanose, and brown spot. Numerous feature sets, such as texture features, three-channel red, green, blue (RGB) features, local binary pattern (LBP) features, and speeded-up robust features (SURF) features, are combined and given as input into the classification layer in the proposed model. We compare our proposed model's performance with existing methods by using several evaluation metrics. The findings demonstrate the SADCNN-ORBM model's superiority in precisely recognizing and classifying citrus illnesses, outperforming all available techniques.

Corresponding Author:

Roheet Bhatnagar
Department of Computer Science, Faculty of Engineering, Manipal University Jaipur
Jaipur-Ajmer Express Highway, Dehmi Kalan, Near GVK Toll Plaza, Jaipur, Rajasthan 303007, India
Email: roheet.bhatnagar@jaipur.manipal.edu

1. INTRODUCTION

Agriculture production is becoming more and more important to the global economy as a result of the world's fast growth in population [1]. Fruits are considered around the world to be the richest in nutrients and cash crops. The agricultural sector started looking for creative methods to increase food production because resources were also running out at the same time. Thus, it allows a researcher to look for novel, extremely fruitful, clear, and efficient ideas. In addition, the fruits come in a variety of sizes, colors, and forms, making it difficult and time-consuming to manually classify and identify diseases in the enormous amount of fruit [2], [3]. And in the coming century, agricultural output will become increasingly significant. On the other side, major obstacles to development include things like climate change, the loss of agricultural land, and new illnesses. According to the report, plant diseases cause around 16% of the world's agricultural losses. citrus huanglongbing (HLB) [4], [5] is reportedly having a major impact on the citrus plantation, according to research issued by the National Research Centre for Citrus. The HLB illness spreads over the entire field through particles in the air as a result of pathogen attacks [6]–[8].

These plants are sensitive to diseases such as citrus canker, greening, and anthracnose [9]. To control these diseases, a large number of chemicals and fungicides are used, which results in environmental pollution and economic losses [10], [11]. Recognizing crop disease in the initial stage is mandatory so that it does not affect production and can also protect farmers from costly spray processes [12], [13]. So, providing a proper diagnosis in a timely and accurate manner is utterly important [14], [15].

To address this, we have provided a thorough study of our self-attention-based dilated convolutional neural network-optimized restricted Boltzmann machine (RBM) model, emphasizing its structure, data preparation methods, and training procedures. We also discussed the dataset, which consists of a wide range of citrus image data utilized for training and assessment. We compared the performance of our proposed model with several cutting-edge deep learning architectures and with conventional machine learning (ML) techniques that are often used for citrus disease diagnosis to show the efficiency and superiority of the proposed model. The rest of the paper is structured as follows: the process for generating and analyzing the speech activity detection self-attention dilated convolutional neural network - optimized restricted Boltzmann machine (SADCNN-ORBM) model is covered in section 2. The experimental findings and discussion are described in section 3. Section 4 concludes the paper.

The contribution of the proposed model lies in its innovative approach to addressing the citrus disease problem in agriculture using artificial intelligence-based technology. The proposed model utilizes cyclic voltammetry (CV) techniques to detect citrus diseases in their early stages to prevent the spread of the disease and minimize its impact on citrus production. The model is designed to combine SADCNN for segmentation and ORBM based on the self-adaptive coati optimization (SACO) algorithm for classification to leverage its strengths, resulting in a more accurate and efficient disease detection model. The SACO algorithm is also utilized to improve the performance of classification by optimizing RBM. The proposed model effectively fuses the proposed model's features to capture diverse and essential information, enhancing its ability to accurately classify citrus diseases. The proposed model classifies citrus diseases into three distinct classes: anthracnose, melanose, and brown spot, which is crucial as different diseases require different management strategies. By providing an accurate and early detection mechanism for citrus diseases, the proposed model can significantly impact agricultural productivity. The proposed model's architecture and use of artificial intelligence (AI) algorithms like SACO allow it to be scalable and adaptable to various agricultural settings and geographical regions.

2. PROPOSED METHOD

Figure 1 depicts the SADCNN-ORBM model's overall system architecture. The illustration shows how the original plant leaf image was originally fed through the preprocessing processes of grayscale conversion, geometric transformation, and image augmentation. The pre-processed image is then segmented using the SADCNN technique. The following stage involves feature extraction for the segmented features collected from the previous stage. In this stage, the features were extracted utilizing speeded-up robust features (SURF), three-channel red, green, blue (RGB), local binary pattern (LBP), and texture features. Concatenated extracted features have been transferred to the ORBM for classification. Here, the SACO algorithm is used to optimize the RBM using two-tier hyperparameter tuning. The following subsections give information on the specifics of how these procedures work.

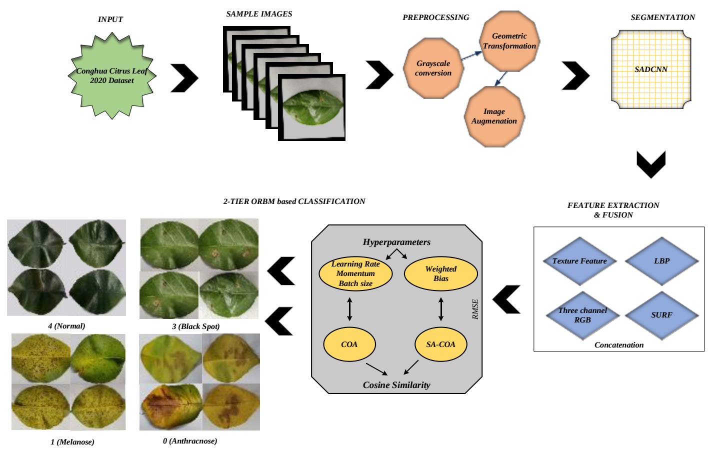


Figure 1. The overall architecture of the proposed mode

2.1. Citrus leaf diseases dataset and image preprocessing

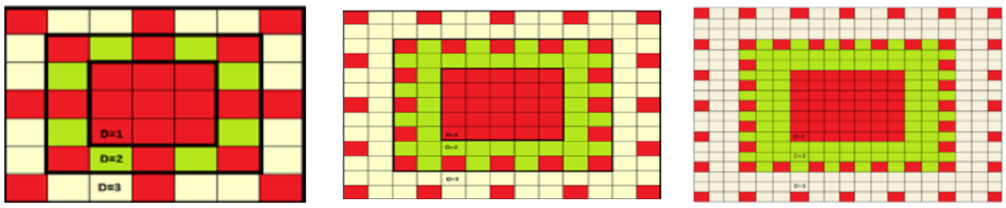
The anthracnose, melanose, and brown spot disease classifications are represented in the Conghua citrus leaf 2020 (CCL'20) dataset [16], along with healthy leaves. It has high-quality images of both healthy and diseased citrus leaves. The initial part of the preprocessing removes any noise present in leaves and subsequently improves the image quality. Pre-processing is done using several procedures to get the data ready for subsequent analysis grayscale conversion, geometric transformation, and image augmentation. First, the image is converted to grayscale, using the grayscale conversion to simplify the data. In addition, it maintains the structural information of the image while lowering its computational complexity. Next, the newly generated grayscale image undergoes modification using a geometric transformation, which alters the unique pixel arrangements in the images by performing operations including rotation, scaling, translation, and flipping. The training dataset is then artificially augmented with the geometrically transformed image using image augmentation to improve the generalization and robustness of the model.

2.2. Segmentation using self-attention-based dilated convolutional neural network

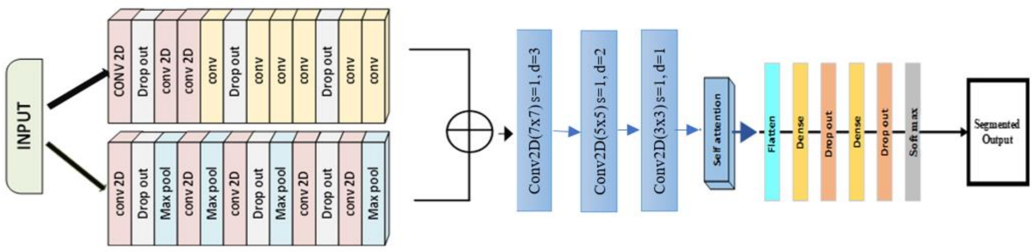
Pooling layers are frequently employed in traditional convolutional neural networks (CNN) to maintain invariance and prevent overfitting; however, they significantly lower the spatial resolution of the input images, erasing the spatial data from the generated feature maps. By dilating the filter before computing the standard convolution, dilated convolution can recover the spatial resolution. CNNs employ dilated convolution, a convolution procedure that allows the network to have a larger receptive field without adding more parameters. It enlarges the kernel by creating voids between the parts that follow one another. The degree to which the kernel is broadened is indicated by a parameter, r (dilation rate or factor). Typically, r-1 spaces are added between kernel parts. The proposed model is built on the 2D-CONV layer, and the dilated CNN easily improves it. The mathematical derivation is presented in (1).

$$y(m,n) = \sum_{i=1}^{m} \sum_{j=i}^{n} x(m + r \times i, n + r \times j)\, w(i,j) \tag{1}$$

where $y(m,n)$ represents O/P, $x(m,n)$ represents I/P, and $w(i,j)$ represents the filter in which m and n are referred to as length and width, respectively. In the dilated convolution layer (1), the variable (r) stands for the dilation rate. When r is used, it gives a distinct number; in this case, "r" is allotted as 1. In this paper, to make a conventional convolution layer a dilated convolution, instead of a convolutional kernel, sparse kernels are used. “Sparse kernel" is nothing but a convolutional kernel with a limited number of non-zero weights, i.e., many of the elements are set to zero. This benefit makes the small size kernel $k \times k$ that has been increased in size, i.e., $K + (k-1)(r-1)$ where r is a dilation rate that might have several values, such as 1, 2, or 3. Consequently, the suggested model possesses a 3×3 filter with a 1, 2, and 3 dilation rates. Similar to the 3×3 filter, the same dilation rates can be added to the 5×5 and 7×7 filters. Figure 2 (a) states the three filter variants (3×3, 5×5, and 7×7) with varying dilation rates. The segmentation architecture using SADCNN is given in Figure 2(b).



(a)



(b)

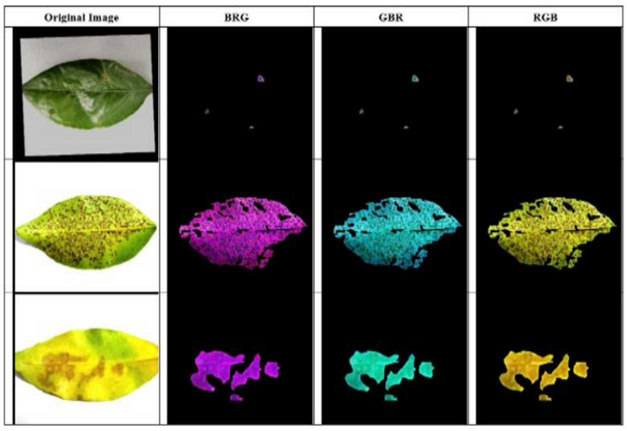
Figure 2. Image segmentation (a) 3×3, 5×5,7×7 filters with dilation rates as 1,2,3 and (b) SADCNN segmentation architecture

2.3. Feature extraction

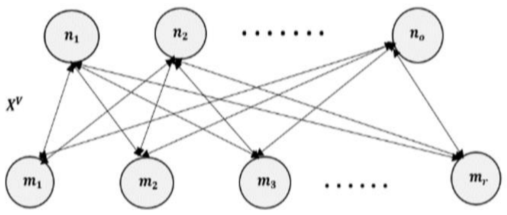
The segmented features obtained from sub section 2.2 are subjected to feature extraction in this step. In this paper, the feature extraction is done in four steps: texture features, three-channel RGB features, LBP, and SURF. Texture feature: the patterns and structures found in the image are described by texture characteristics. We employed the sum of entropy, variance, gradient, and energy as four texture characteristics. The overall amount of disorder or complexity in the texture pattern is represented by the sum of entropy. The sum of variance measures how evenly distributed the data points are throughout the texture image. The sum of the gradient determines variations in pixel intensity within an image and also locates borders or boundaries. The frequency of pixel intensity pairings at various spatial offsets within the image is represented by the sum of energy. Three-channel RGB features: the three channels that make up an RGB image are R, G, and B. As a result, we arrange the three channels differently, such as RGB, GBR, and BRG, as shown in Figure 3(a). LBP: the 3×3 neighborhood was the original application for LBPs. The LBP code of the center pixel is obtained by thresholding the pixel values of its eight neighbors by their value, then summing the resulting thresholder binary values with powers of two. SURF: the SURF detector is used to find structures that resemble blobs. The feature interpretations from each kind are concatenated into a single feature vector as part of the fusion process [17].

2.4. CLASSIFICATION USING OPTIMIZED RESTRICTED BOLTZMANN MACHINE

The extracted features are input into the classification layer, where an ORBM performs the classification task. The optimized unsupervised energy-based generative model is a cutting-edge framework that leverages advanced optimization techniques to effectively learn and generate data in an unsupervised manner. The model's architecture, shown in Figure 3(b), enables intricate interactions through the establishment of connections between neurons in the hidden layers and the visible layer [18]. This is achieved by implementing a two-way interaction mechanism with symmetric weights. The mathematical representation of the energy function F (m, n, θ) is depicted as in (1). RBM is trained using the contrastive divergence (CD) algorithm, an unsupervised method.



(a)



(b)

Figure 3. Image classification (a) extracted three-channel RGB features and (b) structure of an RBM

In RBM, $n = (n_{1,}\, n_2 \ldots.n_o)$ and $v = (m_{1,}\, m_2 \ldots.m_r)$ represent the hidden and visible layer feature vectors, whereas $X^V$is a $i \times j$ weight matrix connecting the neurons. For the parameter set θ ={X^V, e_m, e_n}, the function (2) as,

$$F(m,n,\theta) = -\sum_{a=1}^{p} e_{ma}\, m_a - \sum_{b=1}^{q} e_{nb}\, n_b - \sum_{a=1}^{p} \sum_{b=1}^{q} m_a y_{ab}^{Q} n_b, L\,(m;\theta) = \frac{\sum_n f^{-F(m,n;\theta)}}{\sum_{m,n} f^{-F(m,n;\theta)}} \tag{2}$$

where X^V, e_m and e_n refers a matrix connecting of RBM, visible neurons and hidden biases respectively. The marginal probability is given as $L(m;\,\theta)$ in (1), $L(m;\,\theta)$ the derivative concerning θ should be calculated first. The revised considerations (3) as,

$$\Delta\theta = \frac{\partial\, log\; L(m;\theta)}{\partial\theta}, \frac{\partial\, log\; L(m;\theta)}{\partial\theta} = F_{data}(m_a n_b) \text{ - } F_{model}(m_a n_b),\ \theta_{\tau+1} = \theta_\tau + \eta\ \Delta\ \theta_\tau \tag{3}$$

where η, τ, $F_{model}(m_a n_b)$, $F_{data}(m_a n_b)$ refers a step size learning, iteration index, the resulting model from sampling data, the mean values of original data respectively. The experimental set up on the proposed SADCNN-ORBM is shown in Table 1.

Table 1. Experimental set up

Steps	Procedure
Input	Conghua citrus leaf 2020 (ccl'20) dataset
Output:	Output: segmentation and classification of citrus leaf
Procedure	Split dataset into training and testing for SADCNN-ORBM model
Initialization	Set *batch_size*, epoch, no of hidden layer, pooling, filters
Create layer	input layer
Module 1	*Conv2d←DropOut←MaxPool, Conv2d←Maxpool, Conv2D←Dropout←MaxPool, Conv2D←Dropout, Conv2d←MaxPool*
Module 2	*Conv2D←Dropout, Conv2d←Conv2d, Conv2d←DropOut, Conv2d←Conv2d, Conv2d←DropOut, Conv1d←Conv1*
Concatenation	Concatenate (Module1 & Module2), Conv2d (3×3)← Conv2d (5×5)← Conv2d (7×7)
Self attention	Self-Attention (Conv2d), Dropout←Dense←Dropout←Dense←Flatten
Feature extraction (v)	$V_1 \leftarrow F_v(Texture), V_2 \leftarrow F_v(LBP), V_3 \leftarrow F_v(Surf), V_4 \leftarrow F_v(RGB + GBR + BRG)$ f_{v_n}=Concate(V1+V2+v3+V4)
Classification	Model.add(RBM) ← f_{v_n}, activation function (softmax)
Loss function	RMSE $= \frac{1}{n}\sum_{p=1}^{n} w_p(y_p - \widehat{y_p})^2$, optimizer= SACO algorithm

2.4.1. Self-adaptive coati optimization algorithm

a. Initialization process

Coatis are regarded as components of the population of the coati optimization algorithm (COA) method, a population-based metaheuristic. The equation (4) shows every coati's location within the search region reveals the values for the decision variables.

$$X_c : x_{c,d} = lb_d + r\,.\,(ub_d - lb_d),\ c = 1,2,\ldots,N,\ d = 1,2,\ldots,m, \tag{4}$$

where X_c is the cth coati in the space search is located, $x_{c,d}$ is the jth variable's decision value, N, m, r, lb_d, ub_d refers the no. of coatis, decision-making number factors, arbitrary no. in the range [0,1], the d^{th} decision variable's upper bound and lower bound respectively. In (5), where F and F_c refers to the value of the objective function obtained based on the c^{th} coati and the vector of the obtained objective function respectively. The population matrix:

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_c \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,d} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & ⋰ & \vdots \\ x_{c,1} & \cdots & x_{c,d} & \cdots & x_{c,m} \\ \vdots & ⋰ & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,d} & \cdots & x_{N,m} \end{bmatrix}_{N,m} \quad F = \begin{bmatrix} F_1 \\ \vdots \\ F_c \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_c) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \tag{5}$$

b. Coati optimization algorithm mathematical model

Phase 1: A simulation of the coati's iguana-attacking strategies is used to mimic the initial iteration of the coati updating population in the region search. In this technique, numerous coatis climb to the tree, approach, and surprise the Iguana. While they wait behind a tree, more coatis come together around the iguana when it hits the ground. When the Iguana hits the ground, the coatis attack it and begins to hunt it. By employing this technique, coatis may go to different areas inside the search region. The iguana is considered to be the best member of the population, according to the COA design. In addition, it is believed that half of the coatis move up the tree while the other half waits for the Iguana to fall to the ground. The equation (6), then, grows again where the coatis are as they emerge from the tree.

$$X_{c,d}^{P1} : X_{c,d}^{P1} = x_{c,d} + r \, . \left(Iguana_d - I . X_{i,j}\right), \text{for } i = 1,2,\ldots,\left\lfloor\frac{N}{2}\right\rfloor \text{ and } j = 1,2,\ldots,m. \quad (6)$$

In the search area, the $Iguana$ is thrown to the ground and then set down somewhere at random. Coatis on the ground move in the search space using (7) and (8), which copy this random location. The update process will accept each coati's new position and raise the value of the target function; otherwise, the coati will remain in its previous location. The equation (9) is used to bring back this updating condition for $i = 1, 2, \ldots, N$.

$$Iguana^G : Iguana_d^G = lb_d + r \cdot (ub_d - lb_d),\ j = 1,2,\ldots,m, \quad (7)$$

$$X_{c,d}^{P1} : X_{c,d}^{P1} = \begin{cases} x_{c,d} + r \, . \, \left(Iguana_d^G - I \, . \, X_{c,d}\right), & F_{Iguana^G} < F_{I,} \\ x_{c,d} + r \, . \, (X_{c,d} - Iguana_d^G), & else, \end{cases}$$
$$\text{for } c = \left\lfloor\frac{N}{2}\right\rfloor + 1, \left\lfloor\frac{N}{2}\right\rfloor + 2, \ldots, N \text{ and } d = 1,2,\ldots m \quad (8)$$

$$X_c = \begin{cases} X_c^{P1}, & F_c^{P1}, \\ X_c, & else. \end{cases} \quad (9)$$

Here X_c^{P1} refers to a new position determined for the cth coati, $X_{c,d}^{P1}$ is its j^{th} dimension, F_c^{P1} is its actual value for the function, r is an arbitrary in the range of real number [0, 1], $Iguana$ represents the Iguana's location in search area, which alludes the best member's position, $Iguana_j$ is its jth dimension, I is an integer chosen at random from the group$\{1, 2\}$, $Iguana^G$ is the iguana's generated position randomly on the ground, $Iguana_d^G$ G is its dth dimension, F_{Iguana^G} refers to the value of the floor function and the objective function is $\lfloor\cdot\rfloor$.

Phase 2: The second step, updating the position of coatis in the search space, is mathematically modeled based on the normal behavior of coatis while facing and avoiding attackers. In order to mimic this behavior, a random position is generated near where each coati is placed using (10) and (11).

$$lb_d^{local} = \frac{lb_d}{t}, ub_d^{local} = \frac{ub_d}{t}, \text{ where } t = 1,2,\ldots,T \quad (10)$$

$$X_c^{P2} : X_{c,d}^{P2} = x_{c,d} + (1 - 2r).\left(lb_d^{local} + r.\left(ub_d^{local} - lb_d^{local}\right)\right),\ c = 1,2,\ldots,N,\ d = 1,2,\ldots,m, \quad (11)$$

The position estimated newly is suitable to raise the objective function's value, which the condition replicates using (12).

$$X_c = \begin{cases} X_c^{P2}, & F_c^{P2} < F_i, \\ X_c, & else, \end{cases} \quad (12)$$

According to the second phase of COA, X_c^{P2} is computed a new position for the c^{th} coati, $X_{c,d}^{P2}$ is its d^{th} measurement, F_c^{P2} refers to the function value impartial, r is an integer drawn at random from the range [0, 1], t is the iteration counter, lb_d^{local} and ub_d^{local} are, respectively, the local upper and lower bounds of the d^{th} choice variable, lb_d and ub_d are, respectively, the d^{th} decision variable's lower and upper bounds.

c. Self-adaptation

The self-adaptation process is done in the COA algorithm to reduce the premature convergence that is happening in the local optima. It also creates a balance between the exploitation and exploration phases. Here, in (13) we are using the logistics map in which the chaotic variable λ is iteratively introduced to perturb the best solution [16].

$$X_i^{P1} : x_{i,j}^{P1} = x_{i,j} + r.\lambda * X_i * (1 - X_i); 0 < \lambda < 4 \quad (13)$$

where λ stands for the chaotic control parameter. In the exploitation phase, when a predator attacks, a coati flees the area. Based on (11) of COA, a sine-cosine (14), (15) adaptation is added at a random place close to where each coati is located in order to simulate this behavior.

$$X_i^{P2} : x_i^{P2} = x_{i,j} + A \sin\left(lb_j^{local}\right) \left|ub_j^{local} - lb_j^{local}\right|, A = rand \times \lambda\left(\frac{t}{T}\right) \quad (14)$$

where t and T represent the current and maximum number of iterations, respectively. If the newly computed location raises the value of the goal function, which this condition reproduces by applying COA's (12), then it is suitable.

$$X_i = \begin{cases} x_{i,j} + A\,sin\left(lb_j^{local}\right)\left|ub_j^{local} - lb_j^{local}\right| & rand < 0.5 \\ x_{i,j} + A\,cos\left(lb_j^{local}\right)\left|ub_j^{local} - lb_j^{local}\right| & rand > 0.5 \end{cases} \tag{15}$$

Algorithm 1. Pseudo-code of SA-COA

```
Start SA-COA.
1.  Key the optimization problem info.
2.  Assign the no. of iterations ($T$) and the no of coatis ($N$)
3.  Positions of all coatis are initialized using (4) and the goal function for this
    initial population is evaluated.
4.  For $t = 1{:}T$
5.      Update the location of the iguana depending on where the member best in the
        population is located.
6.      First phase: attaching and hunting strategy on the iguana (Exploration phase)
7.        For $i = 1 + [N/2]{:}N$
8.            Self-adaptation using a chaotic variable: Calculate the latest location for
              the $i$th coati by means of the logistics map in which the chaotic variable $\lambda$ is
              iteratively introduced to perturb the best solution which is stated in (13).
9.            Update the position of the $i$th coati using (9).
10.       End for
11.       For $i = 1 + [N/2]{:}N$
12.           Calculate the random position for the iguana using (7)
13.           Calculate another new location for the $i$th coati using (8).
14.           Using (9), adjust the ith coati's location.
15.       End for
16.     Second phase: getting away from predators (Exploitation Phase)
17.           Calculate the local bounds for variables using (10).
18.       For $i = 1{:}N$
19.           Sine-cosine-based Self-adaptation: Calculate another new for the $i$th coati
              by introducing a sine-cosine factor near the position of each coati at a
              random position using (14).
20.           Update the position of the $i$th coati using (12).
21.       End for
22.     Store the best contender answer found so far.
23.   End for
24.   Output getting best-obtained solution by COA for the given problem.
End SA-COA.
```

2.4.2. Hyperparameter tuning

The purpose of the optimization is to find the optimal parameters to lower the objective or loss function. This objective function is called root mean square error (RMSE) in (16).

$$\text{RMSE} = \frac{1}{n}\sum_{p=1}^{n} w_p\left(y_p - \widehat{y_p}\right)^2 \tag{16}$$

where n signifies the sample of number, y_p represents the sample true value, and $\widehat{y_p}$ denotes the predictive value. In this paper, the SACO algorithm is utilized to improve the performance of classification using RBM. The performance is improved by tuning the hyperparameters. The hyperparameters used for this paper include learning rate, momentum, epochs, batch size, and learning rate. This is done by first transferring the segmented image from the SADCNN to the RBM for fine-tuning. In the RBM, optimization happens by varying the hyperparameters using variants of the coati optimization algorithm (COA) [16]. After this process, the results were evaluated using cosine similarity, which calculates the similarity between them. Then the final results are given to the classification layer. The optimization process done as a result of tuning the parameters is shown in Figure 4.

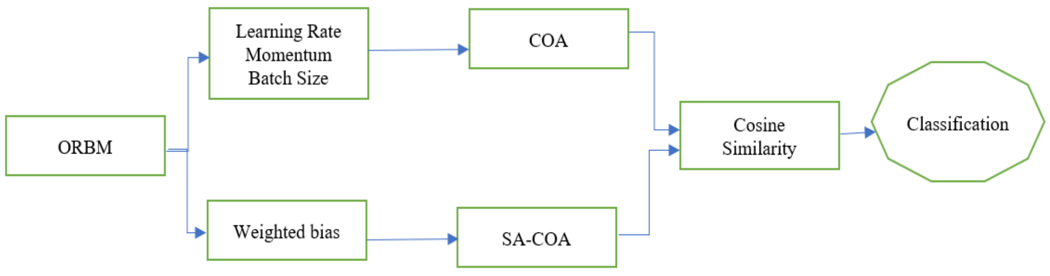


Figure 4. Tuning of the parameters

3. RESULTS AND DISCUSSION

In this section, we compare the performance of the proposed methodology to that of well-known methods like recurrent neural networks (RNN) [18], CNN [19], long short-term memory (LSTM), and mask region-based convolutional neural network (MRCNN). Table 1 presents an analysis of the overall performance of different models based on performance metrics. The proposed model outperforms all the other models by achieving the highest accuracy (0.997685), precision (0.987593), sensitivity (0.987593), specificity (0.987775), F-Measure (0.987593), Matthews correlation coefficient (MCC) (0.975368), negative predictive value (NPV) (0.987775). Also, it has a low rate of misclassifying positive and negative instances, which is indicated by the lowest false positive rate (FPR) (0.012225) and false negative rate (FNR) (0.012407) values. The graphical representation of the results obtained from the proposed model with the other comparative methods is shown in Figure 5.

Table 1. Analysis of the overall performance

Metrics	CNN	RNN	LSTM	MRCNN	Proposed
Accuracy	0.962581	0.944515	0.919289	0.959264	0.997685
Precision	0.965347	0.942643	0.914286	0.970822	0.987593
Sensitivity	0.962963	0.947368	0.916905	0.948187	0.987593
Specificity	0.962162	0.941624	0.921466	0.970667	0.987775
F-Measure	0.964153	0.945000	0.915594	0.959371	0.987593
MCC	0.92502	0.889032	0.838271	0.918796	0.975368
NPV	0.959569	0.946429	0.923885	0.947917	0.987775
FPR	0.037838	0.058376	0.078534	0.029333	0.012225
FNR	0.037037	0.052632	0.083095	0.051813	0.012407

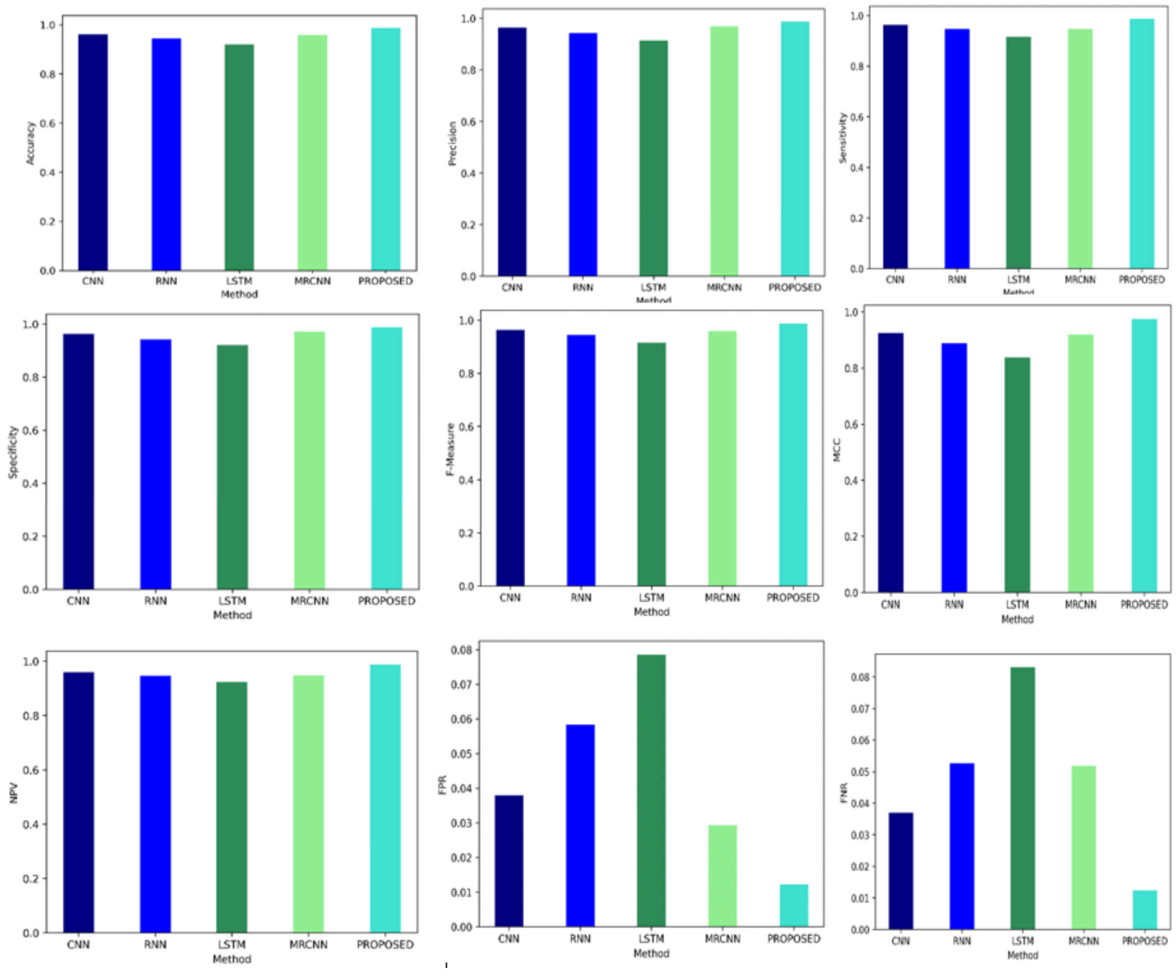


Figure 5. Graphical representation of the metrics

3.1. Comparative analysis

Several distinct methods used to detect plant diseases are compared in Table 2. The suggested SADCNN-ORBM method has the best overall accuracy, surpassing all other current approaches, suggesting its potential as a cutting-edge strategy for precise and effective plant disease diagnosis. The utilized dataset for plant detection diseases is the CCL'20 dataset, which is presented and contrasted in Table 3. The proposed SADCNN-ORBM method exhibits superior overall accuracy, as shown in Figure 6, compared to existing methodologies.

Table 2. Comparison with existing papers

References	Year	Technique	Overall Accuracy in %
Rahman *et al.* [15]	2022	End-to-end anchor-based deep learning model	97.20
Khattak *et al.* [20]	2021	Deep neural network model	95.65
Elaraby *et al.* [21]	2022	Optimization deep learning approach	94.30
Faisal *et al.* [22]	2023	Deep transfer learning based detection and classification	99.58
Dhiman *et al.* [23]	2023	CNN with LSTM and edge computing	98.25
Proposed	2023	SADCNN-ORBM	99.76

Table 3. Dataset comparison with existing papers

References	Year	Dataset	Technique	Accuracy (%)
Dehghani *et al.* [17]	2022	CCL’20	DL-based ORBM	93.75 %
Dai *et al.* [24]	2023	CCL’20	CNN generated DL	97.04%
Pal [25]	2022	CCL’20	OMN-CNN	93.78%
Proposed	2023	CCL’20	SADCNN-ORBM	99.76

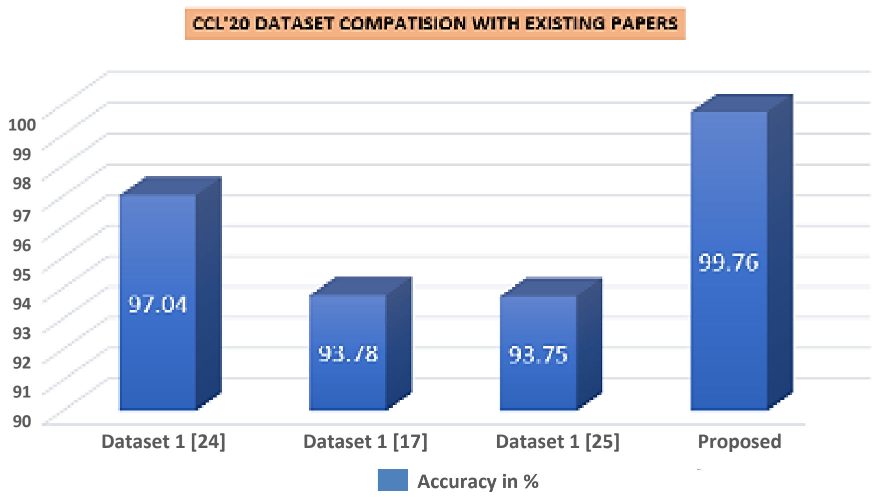


Figure 6. Graphical representation of the CCL’20 dataset

4. CONCLUSION

In conclusion, we have introduced a unique hybrid deep learning model, SADCNN-ORBM, for detecting and classifying citrus diseases utilizing CV-based technologies. Our suggested approach combines the benefits of a SADCNN for disease segmentation with an ORBM driven by the SACO algorithm for disease classification. To improve the model's overall performance, we have also included several feature sets that are combined and supplied into the classification layer. These findings support our model's ability to identify and categorize citrus diseases with 99.8% accuracy. Our concept has a sizable potential influence on agricultural practices. Early disease identification and categorization allow farmers to take the necessary precautions to stop or slow the spread of infections, thereby improving crop yields and overall agricultural production.

REFERENCES

[1] R. Lal, “Home gardening and urban agriculture for advancing food and nutritional security in response to the COVID-19 pandemic,” *Food Security*, vol. 12, no. 4, pp. 871–876, 2020, doi: 10.1007/s12571-020-01058-3.

[2] A. Mehmood, M. Ahmad, and Q. M. Ilyas, "On precision agriculture: enhanced automated fruit disease identification and classification using a new ensemble classification method," *Agriculture (Switzerland)*, vol. 13, no. 2, 2023, doi: 10.3390/agriculture13020500.
[3] H. M. R. Iqbal and A. Hakim, "Classification and grading of harvested mangoes using convolutional neural network," *International Journal of Fruit Science*, vol. 22, no. 1, pp. 95–109, Dec. 2022, doi: 10.1080/15538362.2021.2023069.
[4] B. Kaur, T. Sharma, B. Goyal, and A. Dogra, "A genetic algorithm based feature optimization method for citrus HLB disease detection using machine learning," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, IEEE, Aug. 2020, pp. 1052–1057. doi: 10.1109/ICSSIT48917.2020.9214107.
[5] D. Ghosh *et al.*, "Huanglongbing pandemic: current challenges and emerging management strategies," *Plants*, vol. 12, no. 1, Dec. 2022, doi: 10.3390/plants12010160.
[6] X. Zhang, B. Li, Z. Zhang, Y. Chen, and S. Tian, "Antagonistic yeasts: a promising alternative to chemical fungicides for controlling postharvest decay of fruit," *Journal of Fungi*, vol. 6, no. 3, pp. 1–15, 2020, doi: 10.3390/jof6030158.
[7] W. Albattah, M. Nawaz, A. Javed, M. Masood, and S. Albahli, "A novel deep learning method for detection and classification of plant diseases," *Complex and Intelligent Systems*, vol. 8, no. 1, pp. 507–524, 2022, doi: 10.1007/s40747-021-00536-1.
[8] G. Geetha, S. Samundeswari, G. Saranya, K. Meenakshi, and M. Nithya, "Plant leaf disease classification and detection system using machine learning," *Journal of Physics: Conference Series*, vol. 1712, no. 1, 2020, doi: 10.1088/1742-6596/1712/1/012012.
[9] B. YM, P. VS, N. SV, P. M, and N. S, "Plant disease detection using deep learning and image processing," *International Journal of Electronic Devices and Networking*, vol. 2, no. 2, pp. 13–16, 2021.
[10] T. N. Pham, L. Van Tran, and S. V. T. Dao, "Early disease classification of mango leaves using feed-forward neural network and hybrid metaheuristic feature selection," *IEEE Access*, vol. 8, pp. 189960–189973, 2020, doi: 10.1109/ACCESS.2020.3031914.
[11] T. A. Shaikh, T. Rasool, and F. R. Lone, "Towards leveraging the role of machine learning and artificial intelligence in precision agriculture and smart farming," *Computers and Electronics in Agriculture*, vol. 198, Art. no. 107119, Jul. 2022, doi: 10.1016/j.compag.2022.107119.
[12] T. A. Shaikh, W. A. Mir, T. Rasool, and S. Sofi, "Machine learning for smart agriculture and precision farming: towards making the fields talk," *Archives of Computational Methods in Engineering*, vol. 29, no. 7, pp. 4557–4597, Nov. 2022, doi: 10.1007/s11831-022-09761-4.
[13] H. R. Bukhari *et al.*, "Assessing the impact of segmentation on wheat stripe rust disease classification using computer vision and deep learning," *IEEE Access*, vol. 9, pp. 164986–165004, 2021, doi: 10.1109/ACCESS.2021.3134196.
[14] J. Abdulridha, O. Batuman, and Y. Ampatzidis, "UAV-based remote sensing technique to detect citrus canker disease utilizing hyperspectral imaging and machine learning," *Remote Sensing*, vol. 11, no. 11, Art. no. 1373, Jun. 2019, doi: 10.3390/rs11111373.
[15] S. F. Syed-Ab-Rahman, M. H. Hesamian, and M. Prasad, "Citrus disease detection and classification using end-to-end anchor-based deep learning model," *Applied Intelligence*, vol. 52, no. 1, pp. 927–938, 2022, doi: 10.1007/s10489-021-02452-w.
[16] S. Dananjayan, Y. Tang, J. Zhuang, C. Hou, and S. Luo, "Assessment of state-of-the-art deep learning based citrus disease detection techniques using annotated optical leaf images," *Computers and Electronics in Agriculture*, vol. 193, 2022, doi: 10.1016/j.compag.2021.106658.
[17] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati optimization algorithm: a new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 259, Art. no. 110011, Jan. 2023, doi: 10.1016/j.knosys.2022.110011.
[18] N. Aslan, S. Dogan, and G. O. Koca, "Automated classification of brain diseases using the restricted Boltzmann machine and the generative adversarial network," *Engineering Applications of Artificial Intelligence*, vol. 126, Art. no. 106794, Nov. 2023, doi: 10.1016/j.engappai.2023.106794.
[19] H. Jhun, H.-Y. Park, Y. Hisham, C.-S. Song, and S. Kim, "SARS-CoV-2 Delta (B.1.617.2) variant: a unique T478K mutation in receptor binding motif (RBM) of spike gene," *Immune Network*, vol. 21, no. 5, 2021, doi: 10.4110/in.2021.21.e32.
[20] A. Khattak *et al.*, "Automatic detection of citrus fruit and leaves diseases using deep neural network model," *IEEE Access*, vol. 9, pp. 112942–112954, 2021, doi: 10.1109/ACCESS.2021.3096895.
[21] A. Elaraby, W. Hamdy, and S. Alanazi, "Classification of citrus diseases using optimization deep learning approach," *Computational Intelligence and Neuroscience*, vol. 2022, 2022, doi: 10.1155/2022/9153207.
[22] S. Faisal *et al.*, "Deep transfer learning based detection and classification of citrus plant diseases," *Computers, Materials and Continua*, vol. 76, no. 1, pp. 895–914, 2023, doi: 10.32604/cmc.2023.039781.
[23] P. Dhiman, A. Kaur, Y. Hamid, E. Alabdulkreem, H. Elmannai, and N. Ababneh, "Smart disease detection system for citrus fruits using deep learning with edge computing," *Sustainability (Switzerland)*, vol. 15, no. 5, 2023, doi: 10.3390/su15054576.
[24] Q. Dai *et al.*, "Citrus disease image generation and classification based on improved FastGAN and EfficientNet-B5," *Agronomy*, vol. 13, no. 4, Art. no. 988, Mar. 2023, doi: 10.3390/agronomy13040988.
[25] A. R. Pal, "Classification of pest-infested citrus leaf images using MobileNet V2+ LSTM based hybrid model," MSc Research Project, National College of Ireland, Dublin, 2022.

BIOGRAPHIES OF AUTHORS

Ashok Kumar Saini received the B.E. and M.Tech. degree in computer science and engineering from Rajasthan University in 2006 and Rajasthan Technical University, Kota, Rajastha, India in 2013 respectively. He is pursuing Ph.D. from Manipal University Jaipur. Currently, he is an assistant professor (senior scale) at the Department of Computer Science and Engineering, Manipal University Jaipur. He has experience around 17 years in academics and has published over 20 research papers in reputable conferences and journals. His research interests include machine learning, deep learning, computer vision, artificial intelligence, database, data structure. He can be contacted at email: shksaini@gmail.com.

Roheet Bhatnagar works at Manipal University Jaipur (MUJ) as a full time Professor in the Department of Computer Science and Engineering and has been associated with the Manipal group since June 2008. He has additional responsibility as director of research. He has vast experience spanning around 25 years both in industry as well as academics. He has done his BE in computer science, MTech in remote sensing from Birla Institute of Technology, Mesra, Ranchi. His PhD is in engineering and in the software engineering domain from Sikkim Manipal Institute of Technology (SMIT), He is a senior member - IEEE, life members ISTE, ISRS and has published over 80 research papers in reputed conferences and journals. He can be contacted at email: roheet.bhatnagar@jaipur.manipal.edu.

Devesh Kumar Srivastava has been working in the capacity of professor (Sen. level) at Department of Information Technology, Manipal University Jaipur, Jaipur Rajasthan India since 2012. He has a total of 22 years rich experience of academic, research and administration activities. His research areas are software engineering, ML, DL. He is a professional member of IEEE and senior member of ACM. He chaired 42 technical sessions and addressed 12 keynote/invited talks in the international conferences of IEEE, Elsevier, ACM, and Springer. He published more than 100 articles in the journal and conference proceedings. He supervised 5 PhD scholars. He can be contacted at email: devesh988@yahoo.com.