



# A convolutional neural network-driven computer vision system toward identification of species and maturity stage of medicinal leaves: case studies with Neem, Tulsi and Kalmegh leaves

Gunjan Mukherjee<sup>1</sup> · Bipan Tudu<sup>2</sup> · Arpitam Chatterjee<sup>3</sup>

Accepted: 9 August 2021 / Published online: 27 August 2021

© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2021

## Abstract

Medicinal plants are used to cure different common and chronic diseases in different Asian countries including India. The easy availability and planting possibility make them popular resources for alternative medicinal practices like Ayurveda. These medicinal plants possess proven healing potential without causing any side effects. The biochemical constituents of the medicinal leaves are the fundamental reasons of their healing power which considerably vary with maturity. The existing practices of maturity detection are largely based on chemical analysis of leaves through different instruments which are expensive, invasive in nature and time consuming. This paper reports a computer vision-based system to classify the medicinal leaves along with the corresponding maturity level. The process reported is advantageous in terms of comparatively easier, faster, less expensive and non-invasive operations. The presented system captures the leaf images in controlled illumination and processed leaf images are fed to the convolutional neural network (CNN) architecture-based system for classification of both the type of leaves and maturity stage. The paper also presents application of binary particle swarm optimization ((BPSO) for arriving the values of the CNN hyperparameters. The potential of the system has been assessed with three popular species of medicinal plants, namely Neem, Tulsi and Kalmegh. The classification results were verified with repeated test runs and different standard metrics including tenfold cross-validation method. The paper shows that the presented CNN-driven computer vision framework can provide about 99% classification accuracy for simultaneous prediction of leaf specie and maturity stage. Such significant performance of the presented method can be considered as a potential addition to the existing chemical and instrumental methods.

**Keywords** Medicinal plants · Computer vision · Color features · Binary particle swarm optimization · Deep neural network · Convolutional neural network

## 1 Introduction

Ayurveda (Kumkar et al. 2017) is a popular field of alternative medicine in India and other Asian countries. From ancient time, it has shown its healing power to many human diseases like blood pressure, digestion disorders, blood purification, different skin diseases and many others. Leaves of medicinal plants containing different biochemical constituents can significantly cause healing of diseases and are frequently used in raw or processed form. Plants like Tulsi (*Ocimumtenuiflorum*), Basaka (*Justiciaadhatoda* or *Justiciavasica*), Neem (*Azadirachtaindica*), Sarpagandha (*Rauwolfiaserpentina*) and Kalmegh (*Andrographispaniculata*) are widely used due to their rich contribution toward the remedy of varying human diseases.

---

✉ Arpitam Chatterjee  
arpitamchatterjee@gmail.com;  
arpitam.chatterjee@jadavpuruniversity.in

<sup>1</sup> Department of Master of Computer Applications, Regent Education and Research Foundation, Kolkata, India

<sup>2</sup> Department of Instrumentation and Electronics Engineering, Jadavpur University, Kolkata, India

<sup>3</sup> Department of Printing Engineering, Jadavpur University, Salt Lake Campus, Kolkata 700106, India

The biochemical compositions of the medicinal leave extensively vary in nature, for instance, *Andrographolide* which is a bicyclic *diterpenoidlactone* and the major constituent of Kalmegh that contributes toward treatment of diseases like respiratory tract infection (Verma 2019). Tulsi majorly comprises *oleanolic acid*, *ursolic acid*, *rosmarinic acid*, *eugenol*, *carvacrol*, *linalool*,  $\beta$ -*caryophyllene* (about 8%) and  $\beta$ -*elemene* (c.11.0%) (Upadhyay 2017) and is used as *adaptogen*. Neem is popularly used due to its antiviral, antibacterial, antifungal and anthelmintic properties which are present in Neem leaves due to several chemicals like *nimbin*, *nimbini* and *nimbidin* (Alzohairy 2016).

The amounts of biochemical constituents present in the medicinal leaves considerably vary with the maturity (Attanayake et al. 2019). Experiments were carried internally using UV–Vis spectroscopy (Renuka et al. 2016) as a validation to the fact that the peaks for *Chlorophyll a*, *Chlorophyll b* and *Carotenoids* occur between 400 and 700 nm (Bojovic 2005; Kamble et al. 2015).

Despite the importance of the maturity detection of medicinal plant, it is majorly done manually by the human expert judging maturity of the leaves by virtue of their experiences. The maturity detection is also done using chemical and analytical techniques like gas chromatography (GC), high performance liquid chromatography (HPLC) and other similar methods. These methods provide excellent results, but high processing time, expensiveness and time-consuming sample preparation are some associated limitations. Also most of these techniques, except the near infrared (NIR)-based ones, are difficult to implement in a portable fashion as they are stationed in testing laboratories which limit these techniques to be realized in real-time detection. The portability limitation also limits these techniques toward development of portable handheld device which can be used on-the-spot rather than sending samples to the testing labs which again need specific storage requirements during transition. The motivation of using computer vision in this work is to develop the framework of a portable computer vision system which can be used on-the-spot and even can be realized in the form of mobile apps for value addition at consumer end. The faster and less expensive detection process using computer vision can be widely used for different leaves across different steps for medicinal product manufacturing like picking up leaves at correct maturity stages and sorting of leaves in terms of species and maturity level.

Computer vision-based approach is also advantageous due to their non-invasive nature of operations and has been employed in quality evaluation of different food and agro products (Rafiq et al. 2013; Faridi and Aboonajmi 2017). Computer vision techniques have been applied in the diverse research fields including the medicinal plants. Leaf

area measurement is the major aspect and can be accomplished using the computer vision (Venkataraman and Mangayarkarasi 2016). Leaf vein extraction is another aspect of research used in botanical interpretation of the leaves and studying differences of different leaves (Wilfa et al. 2016). Classification of plant leaves and different plants with the help of different classifiers has become popular research area. Different soft computing approaches have been successfully used for the same. A neuro-fuzzy and neural network-based approach for shape feature selection template based leaf classification has shown about 94% accuracy (Chaki et al. 2015). Particle swarm optimization (PSO)-based classification using GIST and local binary pattern (LBP) features has shown about 98% accuracy (Keivani et al. 2020). An improved accuracy of almost 99% has been achieved using the hybrid of PSO, gray wolf optimization (GWO) and support vector machine (SVM) models (Eid et al. 2018). Almost similar tune of classification has been reported using an extreme learning machine approach where the gray level co-occurrence matrix (GLCM), color and Fourier descriptors have been used as features (Turkoglu et al. 2019). Use of SVM along with adaptive boosting (AdaBoost) has been reported where the morphological features of leaves have been used to classify and have shown a recognition rate of about 95% (Mahajan et al. 2021). Application of CNN has as well been reported for plant leaf recognition and leaf disease detection (Sardogan et al. 2018; Deepalakshmi et al. 2021; Karthik et al. 2020; Jeon et al. 2017). It has been successfully applied for tomato, soya, corn and other plants for early detection of leaf diseases (Karthik et al. 2019; Panigrahi et al. 2020) and in all the cases the resulted recognition accuracy has been reported in the range of 94–98%. To recognize the medicinal plant species, also application of CNN has been reported. Application of CNN in (Bhuiyan et al. 2021) called MediNET resulted about 85% accuracy. In Catur et al. (2020), possibility of a Java-based Android app toward medicinal leaf classification has been reported and shown about 86% accuracy for 5 types of medicinal leaves. In Yuanita et al. (2021), AdaBoost has been used with CNN and reported an accuracy of 91%. The literature survey reveals that the room of exploring the potential of CNN to develop a computer vision system that can simultaneously predict the species and maturity stage of the medicinal plant leaves is still open.

The neural network (NN)-based approaches for classifications have shown the promising results in terms of overall performance and accuracy. The neural network system is made up of number of connected layers, and the model is trained to adjust the connection weights as per the desired output (Costa et al. 2020; Ahmed et al. 2021; Li et al. 2021). Despite the wide spread applications of NN as classifier in the domain of computer vision, it requires good

number of characteristic features of the objects to be classified. Classification of medicinal plants in terms of maturity levels requires number of distinctive characteristic features of the leaves from different types of medicinal plants of respective maturity levels. Extraction of such discriminating features of leaves is time consuming and requires large number of meticulous observation of the variation in their characteristic features. These feature extraction part can be simplified by adopting the deep learning methods, particularly with CNN. The deep learning methods have come up in different variations, namely deep neural network (DNN) (LeCun et al. 2015), deep belief network (DBN) (Hinton 2006), CNN (Jaswal 2014), recurrent neural networks (RNN) (Lipton and Berkowitz 2015) and recursive neural tensor networks (RNTN) (Salle and Villavicencio 2018). All these architectures are based on the machine learning dynamics and exploit many layers of nonlinear information processing, extraction of features with their transformations for pattern analysis and classification in both supervised and unsupervised learning. In this work, a CNN model has been adopted since it needs no hand engineered feature extraction. The cascaded model of CNN as has been implemented can results in faster detection rate. However, CNN consists of many hyperparameters which require to be optimized to get desired accuracy. This paper presents a binary coding of the hyperparameters that are further optimized using BPSO to find the optimum model.

Following the basic flow of computer vision application methods, this work also starts with capturing the images of leaves belonging to different species and different maturity levels. The image capturing is an important operation as illumination variation can cause changes to the features. To avoid the illumination variation, an imaging chamber was developed by the research team which can help to maintain a consistent imaging condition throughout the experiments. The work was realized using plant leaves collected from local plantations and classified by experienced Ayurveda practitioners. The care was taken to ensure that the leaves were collected from the same group of plants identified prior to the experiments to avoid any cross-species samples. The results of classification were repeatedly tested for assessing the performance as well as repeatability of the presented method. The performance was also evaluated using tenfold cross-validation, confusion matrix and other important classification performance metrics. The contribution of the paper is twofold: a BPSO approach toward hyperparameter optimization of CNN and simultaneous classification of leaf species and corresponding maturity stage. The experimental results show that for all the three species under consideration the process can provide about 99% classification accuracy both in terms of identification

of species and maturity level of the subjected medicinal leaves.

## 2 Experiments

### 2.1 Sample collection

The overview of the presented method is shown in Fig. 1 and the individual components of the flow diagram are explained in following sections. In this work, three popular types of medicinal plants were selected due to their ready availability. These trees are frequently planted in Indian household. The samples were collected in the same season to avoid the seasonal effects on the leaves and from the same trees of domestic gardens. Total 1500 leaves for each types of medicinal plant were collected comprising 500 leaves for each classes of maturity. The maturity stages were labeled into three broad classes, namely premature, mature and over-mature. The age of the leaves by which they are graded is varying with the particular plant since the lifespan of the three plants taken in this work largely varies. For example, Neem trees have very long lifespan even more than a century in some cases, while the Tulsi plants can survive for few years, if protected well from seasonal changes especially in rainy seasons. The initial gradation of the sample leaves was done by the human experts comprising of Ayurvedic practitioners. The samples were carried inside the controlled temperature carriers from the plucking point to image acquisition point and subjected to the imaging chamber on the same day.

### 2.2 Image acquisition

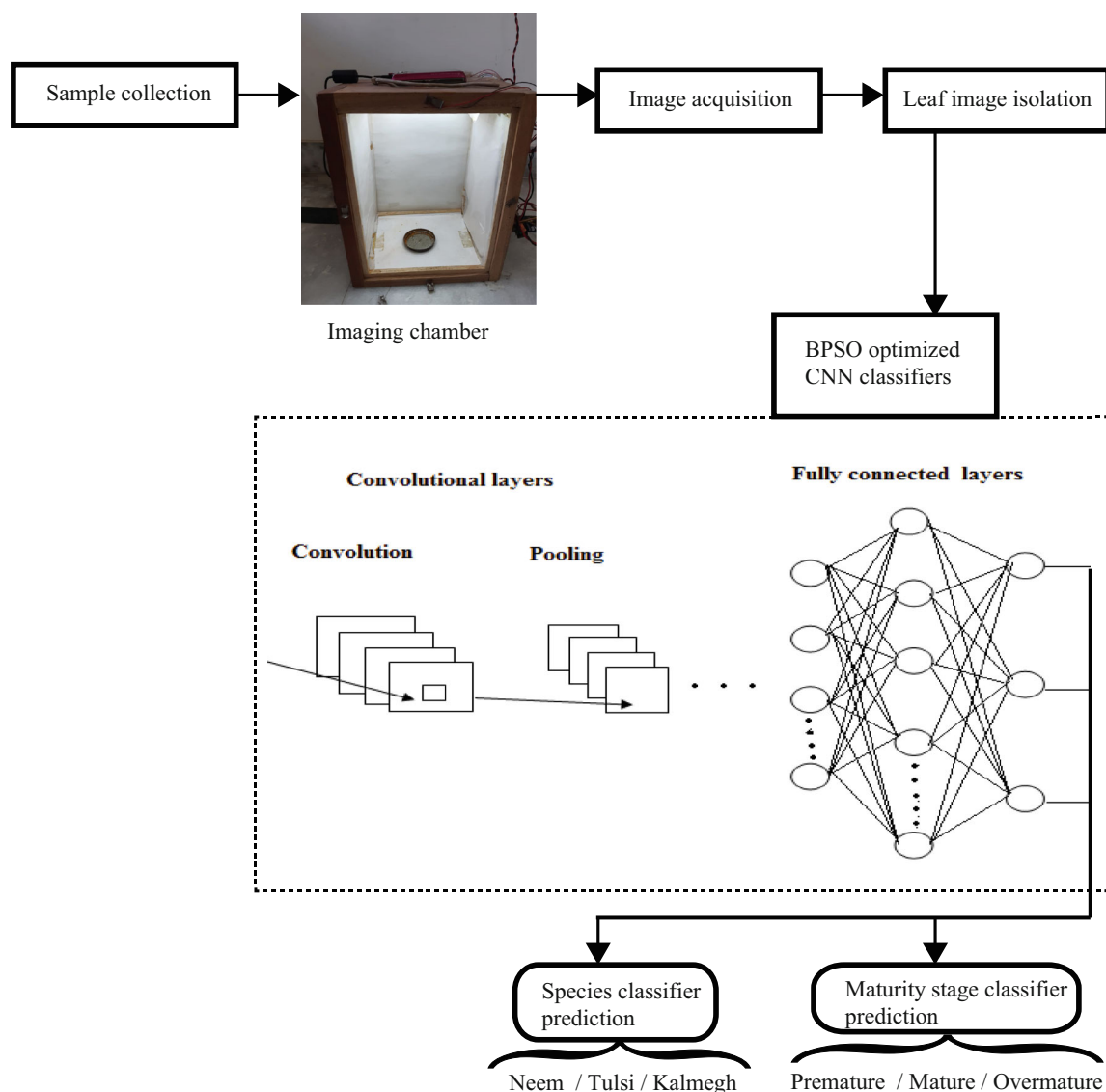
The collected samples as mentioned in the previous section were placed in the image acquisition system. To address the illumination variation, an imaging chamber was made in-house as shown in Fig. 2. The casing of the chamber was made of wood, and the dimension was  $0.5\text{ft} \times 0.5\text{ft} \times 1.5\text{ft}$ . White LED lights were used as the source of illumination which was mounted to the four corners of the chamber. The adjustable base was made to adjust the distance between the camera lens and the object under study for obtaining the desired focus. The chamber was operated at 12 V DC power supply and 20 mA current consumption. Keeping in mind for future real-time development of mobile app and handheld instruments, the images were captured in JPEG format which is lossy in nature but most widely used image format across different devices including mobile phone cameras. At the same time, if the system can perform well with JPEG images, it can be presumed that the system can maintain the performance consistency with other image formats that are lossless and

can provide higher degree of image information. The images were captured without any camera filters, zooming options and external light sources like flash light of the camera. The images were taken using SONY DSC-W30 and interfaced to the computer using USB2.0 connectors.

### 2.3 Leaf isolation

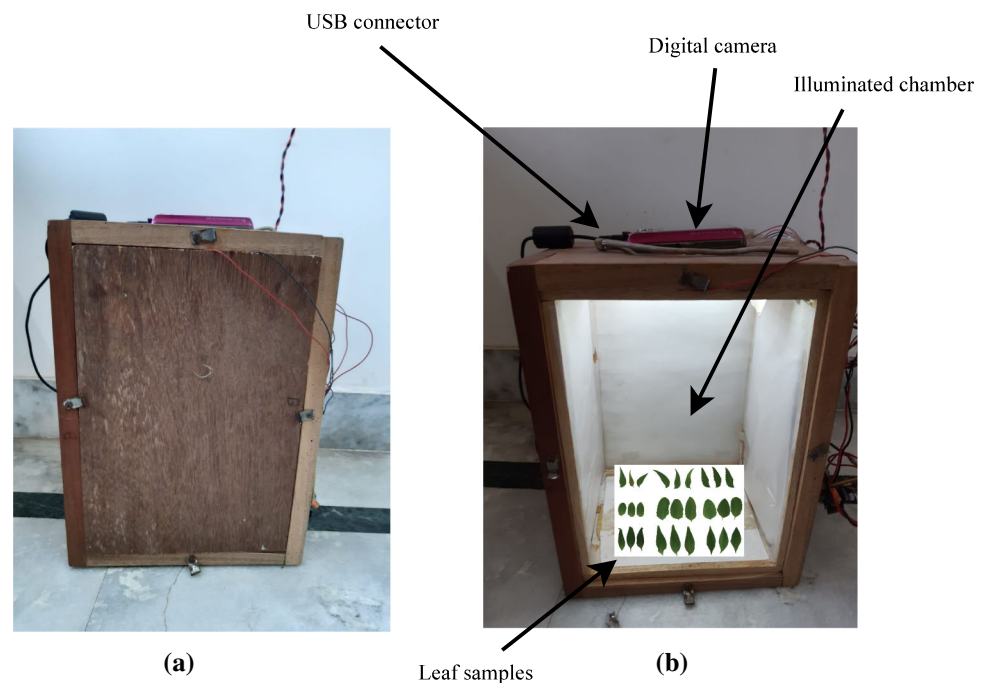
The plant leaves under consideration of this work were small in size; hence, taking image of individual leaves is a tedious job. A leaf isolation algorithm was adopted to address that. This algorithm can isolate individual leaves from the image taken in the imaging chamber laying down multiple leaves on a white paper as shown in Fig. 3a. The algorithm described in the following section counts the number of colored leaves placed on the white sheet of fixed

width followed by placing bounding box around individual image as shown in Fig. 3b and isolation of individual leaf out of the total arrangement as shown in Fig. 3c. Leaves of different species under consideration of this research were arranged on the white sheet of paper in a single row maintaining a fixed gap. A horizontal scan line was chosen crossing all the leaves from left to right. Pixel values were scanned from left to right along the scan line tracking for number of transition value of a particular color channel between two consecutive pixels with respect to a fixed threshold value of 100. The number of leaves were tracked by the number of such transition for each successive pair of consecutive pixels with first one having value more than 100 corresponding to the background and the second one having value less than 100 corresponding to that of the leaf. The two extreme coordinates (left and right) on the scan



**Fig. 1** Schematic process flow diagram of the presented method

**Fig. 2** Indigenous imaging chamber: **a** the imaging mode and **b** sample loading mode



line was measured along the  $x$  axis and the length was divided by number of leaves laid on the white sheet to obtain the bounding box position around each of the leaf. Bounding boxes were laid around each leaf based on coordinate positions. Finally, the image of leaves under each bounding boxes was cropped out.

## 2.4 CNN classifier

In this work, four CNN classifiers were used to classify leaf species and maturity stage of individual leaf species simultaneously. CNN model comprises three operational layers, namely the convolution layer, pooling layer and fully connected layer. The convolution layer and pooling layer play pivotal role in understanding the image information directly from the input images and reduces the critical steps of feature extraction and feature selection as required by many conventional classifiers. The fully connected layer is very similar to the conventional NN architecture with multiple hidden layers and a single output layer holding the number of nodes equal to the number of classes in the problem.

The CNN model development has different major approaches. In some cases, the model is developed from the scratch with different number of layers by using soft computing algorithms to achieve the desired application specific model. It is also been a common practice to start with standard pretrained models like AlexNet, GoogleNet, VGGNet, etc. (Sapiazko 2018) and optimizing the model for the problem specific requirements. The third major model development strategy is starting with a basic model

with defined number of layers based on experience and expertise followed by tuning the hyperparameters to achieve the optimum model. This work adopts the third approach since in our case the experimental dataset is not huge enough like most of the reported works with pretrained models (experimental dataset > 5000 images). Our experimental in-house dataset comprises 4500 images; hence, a comparatively shallow CNN model may produce satisfactory classification accuracy in a faster and less expensive manner. The detailed dynamics of CNN architecture and a binary particle swarm optimization-driven hyperparameter tuning as used in this work have been consolidated in following sections. The schematic representation of the CNN model is shown in Fig. 4.

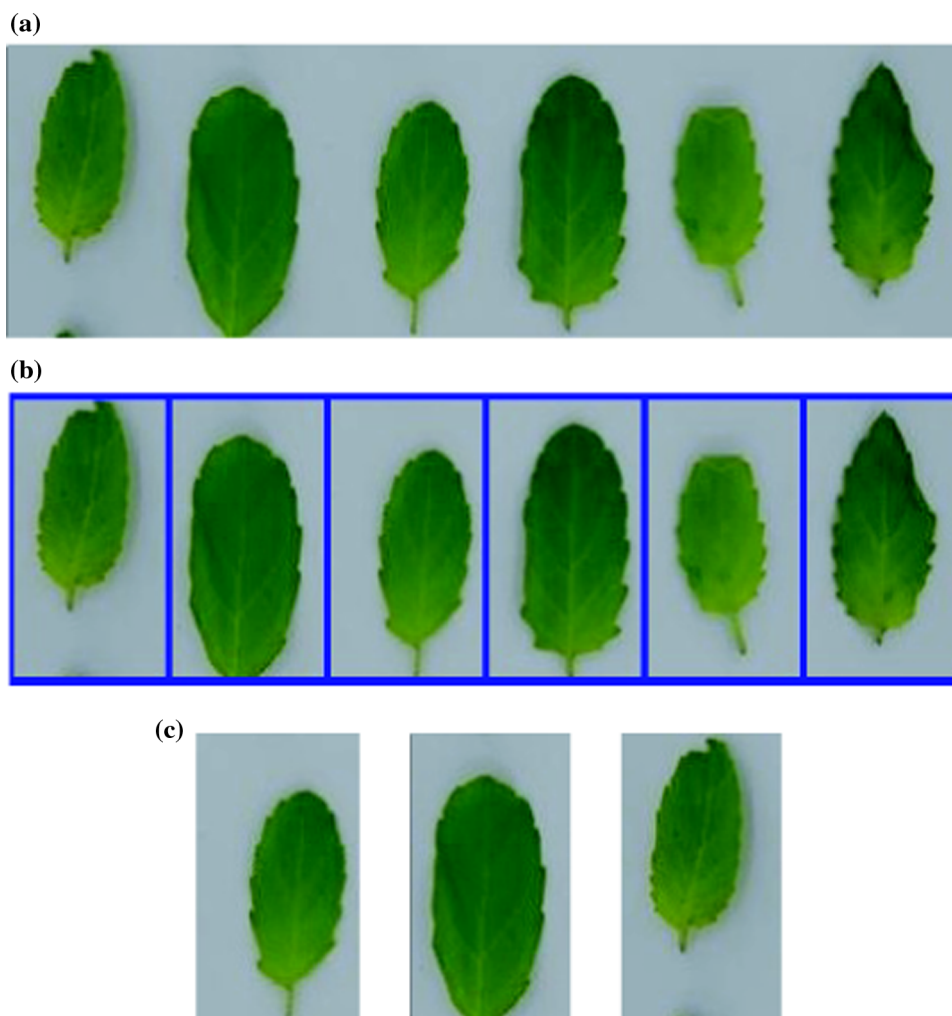
### 2.4.1 Convolution layer

Convolution layer plays the most crucial role in the CNN architecture. It creates a feature map from the input images that are connected to the pooling layer. The feature map is generated using the convolution operation on the input images using a filter bank. Convolution layer includes different hyperparameters including input image size, filter size, number of filters, stride and padding. Stride is the pixel number by which the filter traverse through the image and padding, commonly used as zero padding, is the process to add additional rows and columns to the images to accommodate the filter traversing through the entire image.

The new feature map produced through the convolution process is treated as the output and finally sent through the nonlinear activation function (Yamashita et al. 2018).



**Fig. 3** Leaf isolation example: **a** captured multiple leaves, **b** laying bounding box around each leaf and **c** isolated leaf images



Neurons belonging to the same feature map share the same weights, whereas the neurons belonging to different feature maps have different weights which lead to the extraction of several features at each location. If  $x$  denotes the input image and  $W_k$  denotes the convolutional filter related to the  $k$ th feature map, then the  $k$ th output feature map is expressed as  $Y_k$  following Eq. (1).

$$Y_k = f(W_k * x) \quad (1)$$

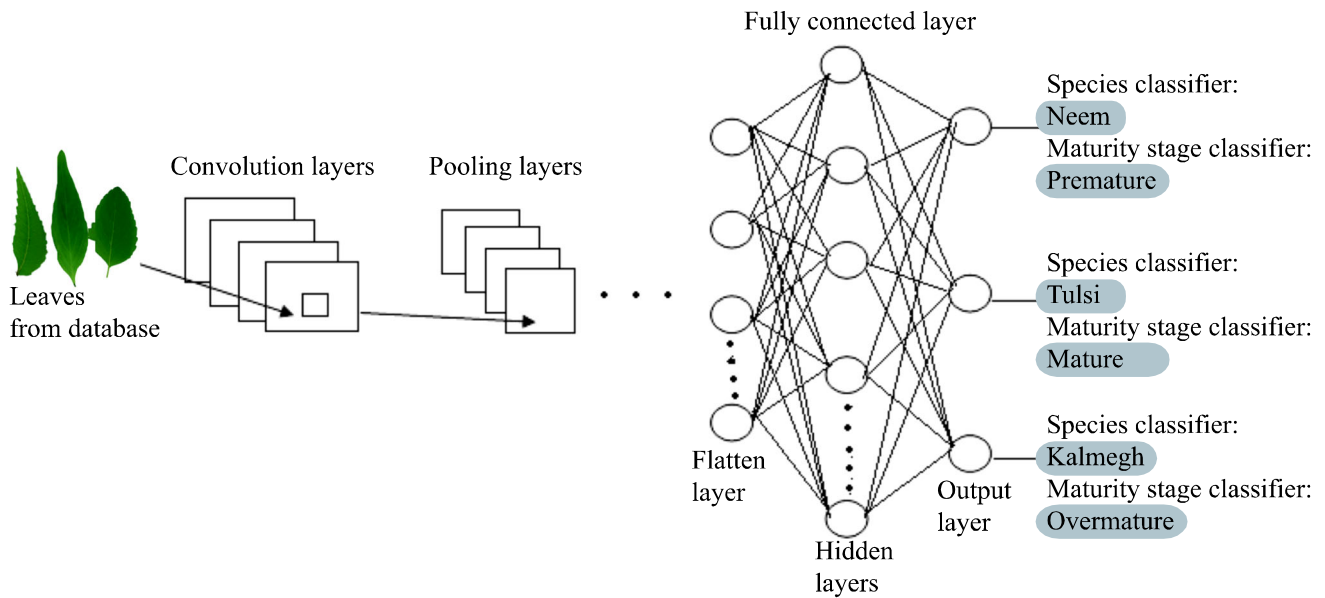
The convolution operator  $*$  computes the inner product of filter model at each location of the input image.  $f(\cdot)$  implies the nonlinear representation of the activation function which extracts the nonlinear features from input images. Sigmoid and hyperbolic tangent functions are some popular nonlinear functions, but rectified linear units (ReLU) are the most commonly used activation function in CNN implementations. The ReLU function is expressed as Eq. (2) which shows that this transform retains the positive values as it is while converting all the negative values to 0. The weights of the kernels in the convolution layers are applied to the input image represented in terms of tensor.

The element-wise product of the kernel and the input tensor is calculated at each location of the tensor and finally getting summed to obtain the output values in the corresponding position of the output tensor.

$$R(z) = \max(0, z) \quad (2)$$

#### 2.4.2 Pooling layer

Convolution layer is followed by pooling layer and is used to reduce the spatial resolution of the feature maps which introduces translation invariance to small shifts and distortions. As a consequence, it reduces the number of parameters to be learned. This layer calculates different moments such as mean and max values within a specified neighborhood region of the convolved image from the convolution layer output. Recent type of pooling model ensures the propagation of maximum value to the next layer with the remaining values being discarded (Traorea et al. 2018). The layer rendering the maximum value is known as the max\_pooling aggregation layer. The global



**Fig. 4** The CNN model for leaf species and maturity stage classification

average pooling is another variation of pooling layer operation which performs the down sampling from the size of height  $\times$  width into an  $(1 \times 1)$  array by means of averaging operation. if  $X_{kpq}$  denotes the element at the  $(p, q)$  location contained in the pooling region and  $R_{ij}$  is defining the receptive field around the location  $(i, j)$ , then the output of the pooling operation on the  $k^{th}$  feature map  $Y_{kij}$  is calculated as Eq. (3).

$$Y_{kij} = \max_{p,q \in R_{ij}} (X_{kpq}) \quad (3)$$

### 2.4.3 Fully connected layers

This is the final hierarchical level of the CNN system which is similar to the general NN architecture. The different hidden layers are connected to each other, and the last hidden layer is connected to the output layer nodes; hence, the term ‘fully connected’ is popularly used. This layer performs the high level reasoning (Traorea et al. 2018). Once the features of any image are extracted in the convolutional layers and down-sampled by the pooling layers, they are flattened using the flatten layer and mapped by the subset of the fully connected layers to the final output of the network. The final output set thus forms the  $N$  dimensional vector where  $N$  denotes the number of classes. The *softmax* operator is used for multiple class problems, and it is expressed as Eq. (4). This operation provides a distributed probability of belongingness among the specified classes and summation of all the probabilities results 1. Hence, the class with highest probability is declared as the class. The nodes of each fully connected layer perform

nonlinear mapping following the activation function dynamics which is ReLU in our case.

$$p(y = j | \Theta^{(i)}) = \frac{e^{\Theta^{(i)}}}{\sum_{j=0}^k e^{\Theta_k^{(i)}}} \quad (4)$$

where,  $p(y = j)$  is the probability of belongingness to  $j^{th}$  class among total  $k$  classes and  $\Theta = \sum_{i=0}^k w_i x_i$ .

### 2.4.4 Training of network

Training the CNN model implies use of learning algorithms to adjust the different parameters in order to achieve the desired network output. The training algorithm adjusts the connection weights based on the calculated loss. The loss function calculates the model performance under a particular kernels and weights by using the training dataset. Kernels and weights are updated through an optimization algorithm like backpropagation and gradient descent. The loss function measures the degree of compatibility among different output predictions. Cross-entropy and mean squared error (MSE) are some of the popularly used loss functions. Gradient is the partial derivative of the loss with respect to each learnable parameter. The parameter update occurs following Eq. (5).

$$w = w - \alpha * \frac{\partial L}{\partial w} \quad (5)$$

where  $\alpha$  is the learning rate and  $L$  and  $w$  stand for the loss function and each learnable parameter, respectively.

### 2.4.5 Hyperparameter optimization using binary particle swarm optimization (BPSO)

Although CNN is a powerful mechanism, the optimum performance needs optimization of the model as there are different hyperparameters which directly affect the performance of CNN. In this work, the hyperparameters of the initial layers were optimized using binary particle swarm optimization (BPSO). PSO is one of the most commonly used iterative search-based optimization techniques across diverse fields including CNN parameter tuning (Wang et al. 2019a, b). It has numbers of proven advantages like simplicity, lesser parameter tuning and higher repeatability (Wang et al. 2019a, b). However, in most of the reported cases the continuous domain PSO has been adopted. In this work, the scope of BPSO has been explored. To implement BPSO, the possible ranges for each hyper parameter were encoded in binary form. Examples of the hyperparameters involved in this study are listed in Table 1. Like PSO, BPSO initializes the search with specified number ( $n$ ) of randomly generated solutions ( $p_i \in P, i = 1, 2, \dots, n$ ). At the completion of individual iteration, the local best ( $p_{\text{best}}$ ) and global best ( $g_{\text{best}}$ ) solutions are obtained. The fitness values of the solutions are calculated using formulated cost function which is commonly an error function subjected to minimization.

The velocity ( $v_i^{t+1}$ ) and position ( $x_i^{t+1}$ ) of  $i$ th solution at  $t + 1$ th iteration are updated following Eqs. (6) and (7), respectively.

$$v_i^{t+1} = \varpi v_i^t + c_1 r_1 (p_{i,\text{best}}^t - p_i^t) + c_2 r_2 (g_{\text{best}}^t - p_i^t) \quad (6)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (7)$$

where  $\varpi$  is the inertia used toward accelerating the convergence and generally initiated at 0.9 to eventually decreased at 0.4 as the algorithm approaches convergence. The learning parameters  $c_1, c_2$  are positive constants generally selected in the range (1, 2). The diversity parameters  $r_1, r_2$  are random numbers in (0, 1) with uniform distribution. In continuous domain, the velocity boundary  $[-v_{\text{max}}, v_{\text{max}}]$  is defined to ensure the swarms are not flying out of the solution space. However, BPSO does not require such boundary handling since the probabilistic mapping of velocity as shown in Eq. (8) maps all the velocity values in the range (0, 1).

$$s_{ij}^t = \frac{1}{1 + \exp(-v_{ij}^t)} \quad (8)$$

where  $i$  and  $j$  are the solution and bit index, respectively. Since the solutions are binary coded, the probability  $s_{ij}^t$  conveys the possibility of changing a bit from its current state, i.e., '1' to '0' or otherwise. Finally, the position updates are performed using Eq. (9) where  $q_{ij}^t$  is a random number generated in (0, 1).

$$x_{ij}^{t+1} = \begin{cases} 1, & \text{if } q_{ij}^t < s_{ij}^t \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

The lost function as expressed in Eq. (10) (Rere et al. 2016) has been adopted as the *cost function* for this work.

**Table 1** Examples of binary encoding of the hyperparameters

Layer type	Variables	Range	Bit length	Encoded example
Convolution	Image size	{32,64,128,256}	2	00 (for 32)
	Im_row = im_column			
	No. of kernels	[4,128]	7	1,000,000 (64)
	Kernel size	[1,8]	3	110 (6)
	Stride size	[1,4]	2	00 (1)
	Total bit length		<b>14</b>	
Pooling	Kernel size	[1,8]	3	100 (4)
	Stride size	[1,4]	2	01 (2)
	Method	{0,1}	1	1 (1)
	maxpooling-0 meanpooling-1			
	Total bit length		<b>6</b>	
Dense (fully connected layer)	No. of nodes	[64,1024]	10	1,100,100,000 (800)
	Total bit length		<b>10</b>	
	Model bit length		<b>30</b>	

Bold values highlight the total length of code for each layer



At individual iteration, the CNN performance was evaluated using the subjected training set with the *gbest* solution of that iteration.

$$O = \frac{1}{2} \left( \frac{\sum_N (y - \hat{y})^2}{N} \right)^{1/2} \quad (10)$$

where  $y$  and  $\hat{y}$  are the expected and actual classification accuracies of the model over  $N$  number of training samples. The algorithm was iterated till the stopping criterion was met which was either of exhausting the defined number of iterations or achieving the desired value of  $O$  that is 0 in our case. An example of the BPSO dynamics is illustrated in Fig. 5.

## 2.5 Model performance evaluation

Confusion matrix is a popular tool that is often used to assess the classification potential. This matrix portrays the important classification statistics in a single representation. The statistical measures include accuracy, sensitivity, precision and specificity. From confusion matrix 4 important measures, namely, true positive (TP), true negative (TN), false positive (FP) and false negative (FN) are

derived (Santra and Christy 2012). These four measures are further extended to calculate precision, recall, overall accuracy, FMeasure, corresponding micro- and weighted averages as shown in Eqs. (11–24). In this connection, receiver operating characteristic (ROC) curve (Fawcett 2006) is used to depict the performance of the classifier for varying threshold values. This curve is constructed by plotting the true positive rate (TPR) termed also as sensitivity recall or probability of detection against the true negative rate (TNR) also termed as the fallout or probability of false alarm.

$$\text{TPR (Sensitivity/Recall)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (11)$$

$$\text{TNR (Specificity)} = \frac{\text{TN}}{\text{TN} + \text{FP}} \quad (12)$$

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \quad (13)$$

$$\text{FNR} = \frac{\text{FN}}{\text{TP} + \text{FN}} = 1 - \text{TPR} \quad (14)$$

$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}} = 1 - \text{TNR} \quad (15)$$

### Encoding example

Convolution			Pooling			Fully connected layer		Layers Binary codes Decimal values Parameters
01	0111000	100	01	101	00	0	1110000100	
128	56	4	2	4	1	0	900	
Im_size	#Kernel	Kernel size	Stride size	Kernel size	Stride size	Max pooling	No. of nodes	

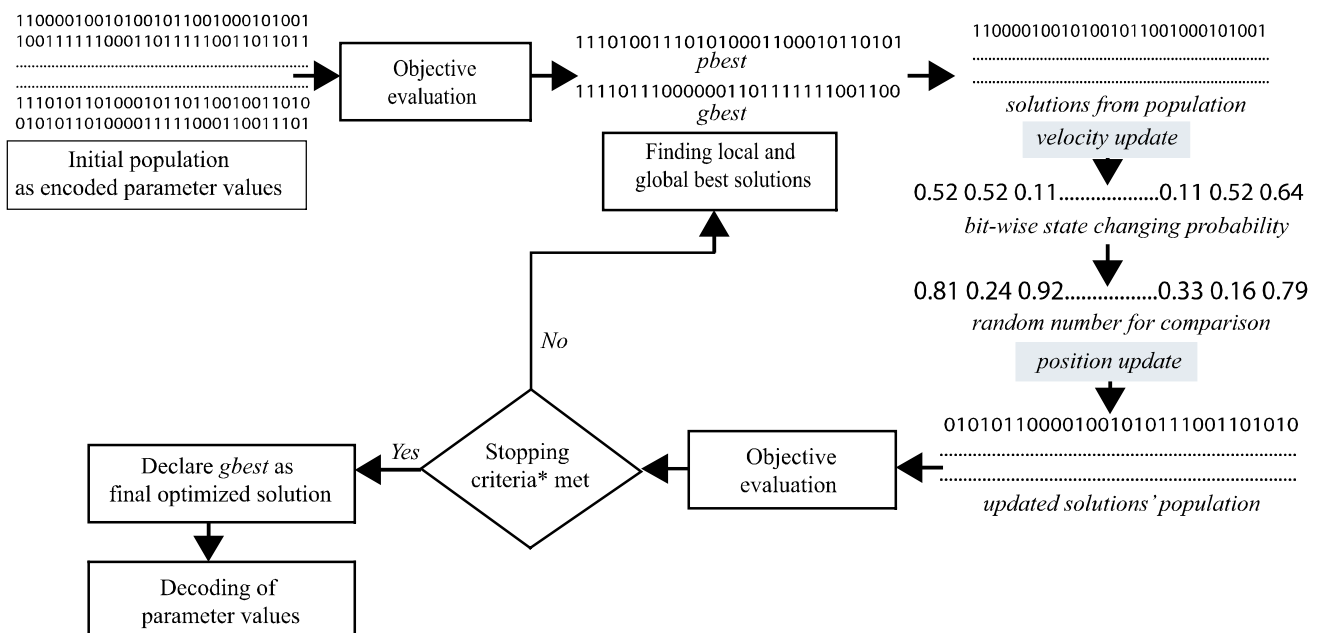


Fig. 5 Schematic representation of BPSO optimization process with binary encoded solutions

$$ACC = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (16)$$

$$FMeasure = 2 \times \frac{(\text{precision} \times \text{recall})}{(\text{precision} + \text{recall})} \quad (17)$$

$$\text{Micro\_average (precision)} = \frac{\sum_c TP_c}{\left( \sum_c TP_c + \sum_c FP_c \right)} \quad (18)$$

$$\text{Micro\_average (recall)} = \frac{\sum_c TP_c}{\left( \sum_c TP_c + \sum_c FN_c \right)} \quad (19)$$

$$\begin{aligned} \text{Micro\_average (FMeasure)} &= 2 \\ &\times \frac{(\text{Micro\_average(precision)}) \times (\text{Micro\_average (recall)})}{(\text{Micro\_average(precision)}) + (\text{Micro\_average (recall)})} \end{aligned} \quad (20)$$

$$\text{Macro\_average (precision)} = \frac{\sum_c P_c}{C} \quad (21)$$

$$\text{Macro\_average (recall)} = \frac{\sum_c R_c}{C} \quad (22)$$

where  $TP_c$ ,  $FP_c$  and  $FN_c$  represent true positive, false positive and false negative for class  $c$  while  $c$  implies the class levels.

$$\begin{aligned} \text{Macro\_average (FMeasure)} &= 2 \\ &\times \frac{(\text{Macro\_average(precision)}) \times (\text{Macro\_average (recall)})}{(\text{Macro\_average(precision)}) + (\text{Macro\_average (recall)})} \end{aligned} \quad (23)$$

$$\text{Weighted\_Average} = \sum_i^N w_i x_i \quad (24)$$

where  $x_i$  represents the  $i$ th item and  $w_i$  represents the relative weight (%) corresponding to the  $i$ th item.  $N$  represents the total number of items.

### 3 Results and discussion

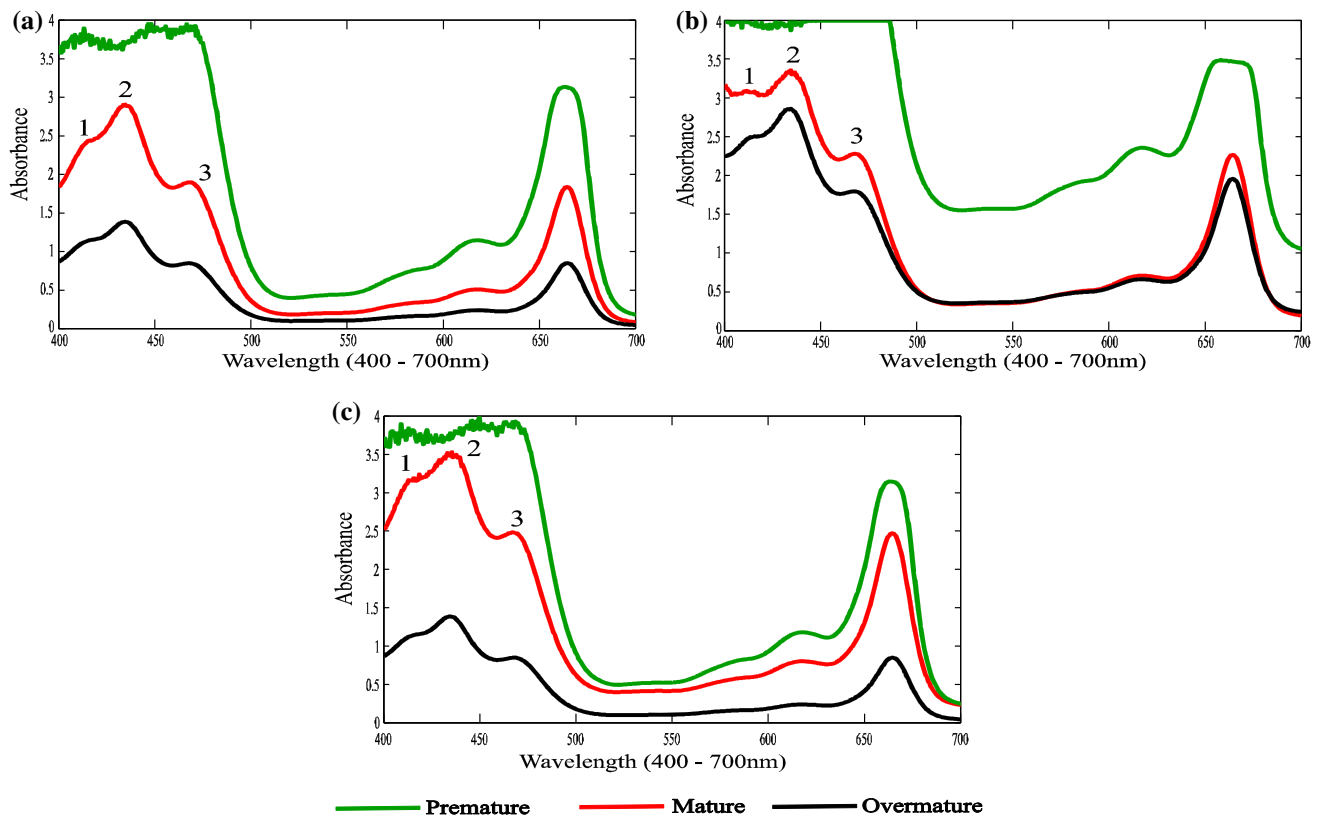
The results of UV–Vis spectroscopy studies are shown in Fig. 6. The wavelength vs. absorption plot in Fig. 6 shows that the peaks for *Chlorophyll a*, *Chlorophyll b* and *Carotenoids* which occur between 400 and 700 nm. It is clearly visible in the plots that the peaks are varying with maturity which indicates the variation in containment of important biochemical constituents of the leaves with maturity.

The implementations of the CNN models were made in Python platform Windows PC environment with the scikitlearn (Buitinck et al. 2013), keras (Chollet 2015) and

TensorFlow backend (Abadi et al. 2016) libraries. The model hyperparameter evaluations were performed using a balanced subset of the entire dataset consisting 1350 observations (30% of the entire dataset). An example of convergence characteristic resulted during BPSO hyperparameter tuning for species classifier CNN is shown in Fig. 7a for 100 iterations. It shows that the model converges to the tune of 96% validation accuracy. This accuracy is expected to increase, while the model is subjected to the entire dataset since higher number of observations can increase the generalization potential of the model. The final model architecture is shown in Fig. 7b, and the example of feature extraction by each layer is illustrated in Fig. 7c. The left most figure of 7(c) shows the background subtracted input image followed by the feature extractions of 3 convolution layers and associated pooling layers. Figure 7c shows that the CNN extracts the deeper and abstract features from the input images as it proceeds to the deeper layers of the model. The final hyperparameter settings for the species classifier are consolidated in Table 2 as an example.

The entire experimental dataset was divided into train, validation and test set in 60:20:20 ratios using *keras* library functions. Total 4 CNN models were implemented in this work, 1 for species classification and 3 for species-wise maturity classification. Hence the working principle was first identifying the specie and then subjecting the features under processing to the maturity classifier of the identified leaf specie. The CNN model was verified using loss and accuracy plots with train and validation sets. The model validation plots with the training and validation datasets are shown in Fig. 8. Figure 8 shows the accuracy and loss plot for 50 epochs. It shows that for all the cases the training accuracy is almost 100%, while the validation accuracy is about 98%. The plots also show the avoidance of overfitting as the plots of training and validation are following the same trend and maintaining narrow gap between them. Similarly the plot of loss and accuracy values resulted by maturity classification CNN for individual species are shown in Figs. 9 and 10, respectively. Both the loss and accuracy plots for the maturity classifiers show that the epoch-wise plots for train and test are not causing visible gap. In all the cases, the training accuracy has reached almost 99%, while the testing accuracy is about 98%. This confirms the generalization capability of the model which in turn confirms the performance potential with unknown test observations.

The models were further assessed with tenfold cross-validation (Berrar 2019) which can reflect the consistency of the model in repeated runs with different train and test data and also the average accuracy rate that the model can offer. This is a popular tool to arrive into the final model. This validation method also ensures that each observation



**Fig. 6** UV-Vis spectroscopy for **a** Kalmegh, **b** Neem and **c** Tulsi leaves. In all the figures, the peaks are labeled 1, 2 and 3 to indicate the concentrations of *Chlorophyll a*, *Chlorophyll b* and *Carotenoids*.

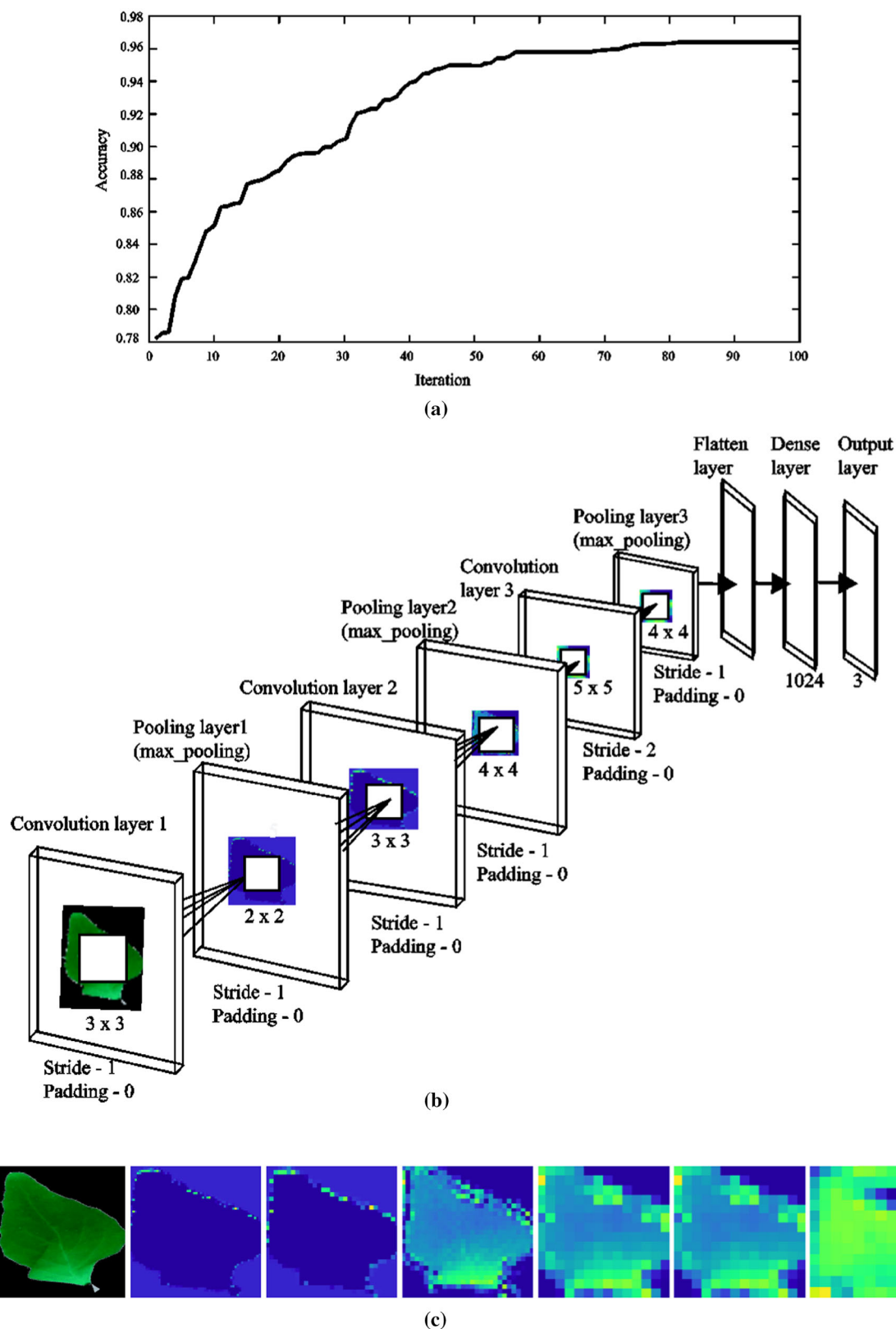
It is clearly visible from the plots that the peaks are varying with the maturity which also indicates the variation of important biochemical of the leaf with maturity

is included as training and testing observation at least once. The accuracy and loss values with the training and testing datasets for leaf species classifying CNN are shown in Table 3. Table 3 shows that the training accuracy is 99.3% and test accuracy is 98.4% which ensures that the model is well generalized. The standard deviation (std. dev.) values for train and test set are  $\pm 0.662\%$  and  $\pm 0.417\%$ , respectively, which is visibly small and ensures the consistency and repeatability of the model. In terms of loss values also the average test loss is as low as the training loss value which again confirms the efficiency of the model in terms of generalization. The average classification accuracy with tenfold cross-validation for maturity stage classification of different leaf species of CNN model is consolidated in Table 4. It shows that for Neem leaves the CNN model can provide 98.8% classification accuracy which is almost same as the accuracy resulted with the training set. Similarly the loss value on test is also equivalently small as that of training set. For Tulsi and Kalmegh leaves, the maturity classifier has resulted 98.3% and 97.7% classification accuracy, respectively. The training set classification accuracy for both of those two species is almost 99%. In terms of loss values, the loss on testing set is consistently low and in parity with the training loss

values. For instance, in case of Kalmegh leaves the average training loss value is 0.022, while the average testing loss value is 0.034 which shows that the model can work well on the unknown test data. It is also important to note that the std. dev. values of accuracy and loss are considerably low which ensure the repeatability and stability of the model.

The confusion matrices for different classifiers are presented in Fig. 11. The consolidated mean performance measures of the confusion matrices are presented in Table 5 for classification of different leaf types and in Table 6 for classification of different maturity grades in respective leaves. The performance measure of the confusion matrix has been expressed in terms of precision, recall, FMeasure, sensitivity and specificity.

Tables 5 and 6 along with the confusion matrices shown in Fig. 11 clearly show the significant potential of the presented model in terms of precision, recall and FMeasure values for classification. Sensitivity and specificity values in all the cases are 98% and above which vouch the satisfactory potential of the model with low error rate in the classification process. Figure 11 also shows the visible potential of the classifiers. The confusion matrix reveals that all the classifiers can classify leaves' maturity with



**Fig. 7** The species classifier CNN model representation; **a** convergence of BPSO process, **b** the optimized model architecture and **c** the feature extraction at each layer of CNN model

99% accuracy, and in many cases it has even reached 100% accuracy. For the species classifier, the ability of correctly identifying Kalmegh is 100%, while it is about 99% for

**Table 2** The hyperparameter setting of optimized species classifier CNN model

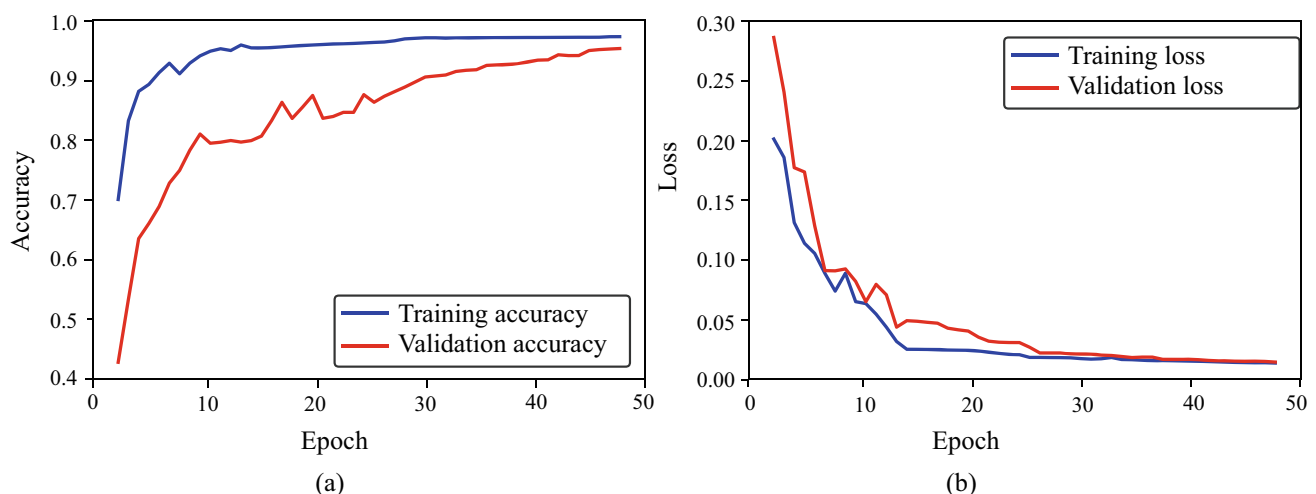
Layer types	Hyperparameters	Parameter values
Convolution 1	Image size	128 × 128
	No. of kernels	32
	Kernel size	3 × 3
	Stride size	1
Pooling 1	Kernel size	2
	Stride size	1
	Method	0
Convolution 2	Image size	128 × 128
	No. of kernels	64
	Kernel size	3 × 3
	Stride size	1
Pooling 2	Kernel size	4
	Stride size	1
	Method	0
Convolution 3	Image size	128 × 128
	No. of kernels	128
	Kernel size	5 × 5
	Stride size	2
Pooling 1	Kernel size	4
	Stride size	1
	Method	0
Flatten	Dimension	6
Dense	No of nodes	1024
Output	No. of nodes	3

Neem and Tulsi leaves. Such high degree of accuracy across all the species and maturity levels can be considered as acceptably good performance.

Cohen Kappa coefficient (McHugh 2012) is another important statistical measurement used in order to assess the inter rater reliability of different classes. The high value of this coefficient marks the good feature selection criterion for the classifier. Tables 7 show the evaluated coefficient measurement for classification of Kalmegh, Neem and Tulsi leaves and for maturity wise classes of as well.

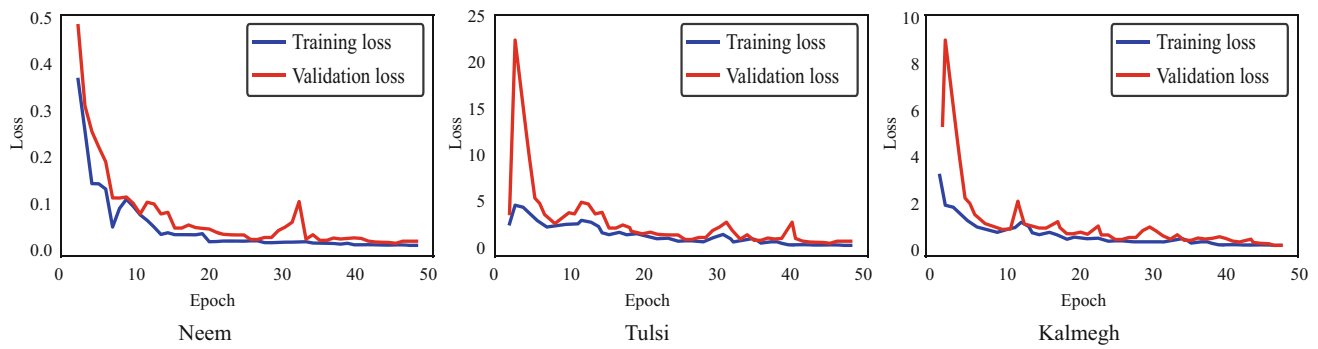
The high values of the Cohen Kappa scores for all CNN classifiers imply the high accuracy performance of the model toward the classification process. As it can be seen for all the cases, the scores are equal or more than 99% that not only ensures significant classification performance of the model but also confirms the goodness of feature selection by the CNN model with the chosen hyperparameters.

The receiver operating characteristic (ROC) curve is the graphical display of plotting for sensitivity or true positive rate (TPR) vs false positive rate (FPR). The area under the curve implies the effective measure of sensitivity and specificity for assessing the inherent validity of diagnostic test. The ROC plot along with the area under the curve of ROC (ROC-AUC) value conveys better potential in terms of higher ROC-AUC values. It is popularly represented against a no-skill or naïve classifier. The ROC-AUC plots are shown in Fig. 12 along with the corresponding values. Figure 12a corresponds to the leaf species classifiers assessment, and it shows that for all the leave species it results in more than 0.98 area of coverage for Kalmegh, Neem and Tulsi. The figure conveys the high potential of correct classification of Kalmegh and Neem leaves, however with a comparatively lower performance for Kalmegh leaves. Figure 12b–d shows the ROC-AUC for different

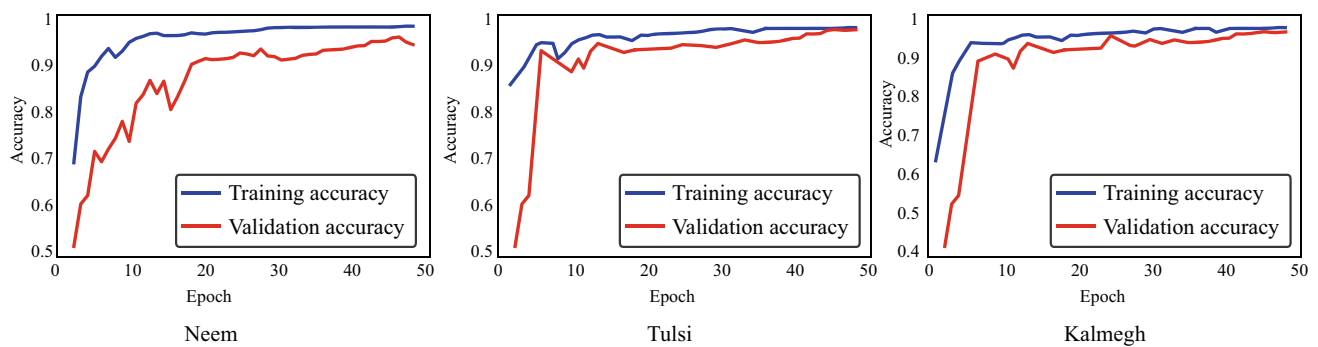


**Fig. 8** The training and validation set accuracy plots for CNN model of species classification **a** accuracy and **b** loss values





**Fig. 9** The loss values with training and validation datasets for maturity classification CNNs for different leaf species



**Fig. 10** The accuracy values with training and validation datasets for maturity classification CNNs for different leaf species

**Table 3** Accuracy and loss values of training and testing data for species classification of Neem, Tulsi and Kalmegh leaves using tenfold cross-validation method

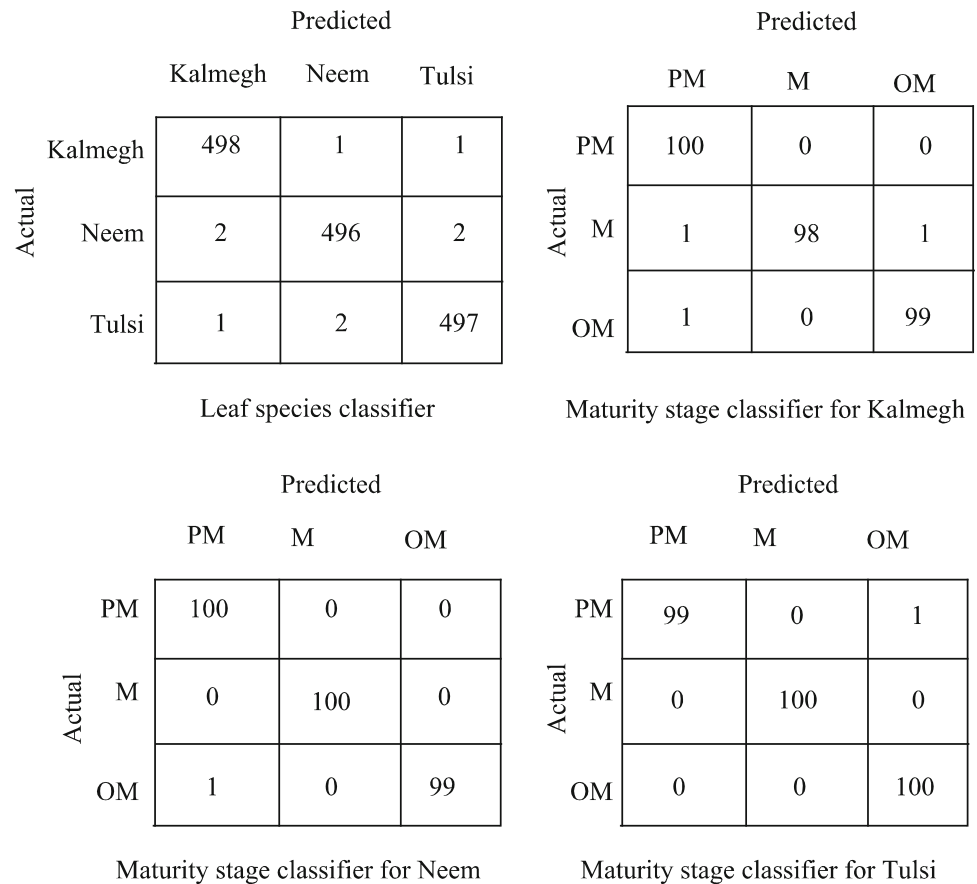
Number of folds	Training data		Test data	
	Loss	Accuracy (%)	Loss	Accuracy (%)
1	0.0114	98.31	0.0304	98.28
2	0.0319	99.94	0.0180	97.75
3	0.0154	99.91	0.0307	98.62
4	0.0072	98.97	0.0347	98.06
5	0.0313	99.60	0.0164	99.07
6	0.0338	98.28	0.0125	98.87
7	0.0157	98.84	0.0192	98.75
8	0.0048	99.83	0.0201	98.10
9	0.0097	99.58	0.0240	98.24
10	0.0147	99.91	0.0271	98.17
Average	0.018	99.32	0.023	98.39
Maximum	0.034	99.91	0.035	99.07
Minimum	0.005	98.31	0.013	97.75
Std. Dev	± 0.011	± 0.662	± 0.007	± 0.417

maturity classes for all the three species where it can be seen that in all the cases the coverage value is more than 0.98 and in some cases it has given value of 1 which indicates all the maturity classifiers can result more than 98% correct maturity stage prediction and in some cases it can even results 100% correct prediction of maturity stage.

Finally, Tables 8 and 9 present examples of the actual model outcome pattern for test leaves and comparison of the performance of presented CNN models with two popular classifiers, namely, ANN classifier using backpropagation multilayer perceptron (BPMLP) modality and SVM. For all these classifiers, a separate color feature extraction

**Table 4** Accuracy and loss values of training and testing data for classification of maturity levels for the leaves using tenfold cross-validation method

Species	Neem		Kalmegh		Tulsi	
	Training	Testing	Training	Testing	Training	Testing
Average accuracy (%)	98.90	98.50	98.80	97.70	99.10	98.30
Average loss	0.050	0.108	0.022	0.034	0.091	0.068
Std. dev. of accuracy (%)	± 0.57	± 0.50	± 0.79	± 0.60	± 0.90	± 0.56
Std. dev. of loss	± 0.030	± 0.183	± 0.046	± 0.019	± 0.077	± 0.048

**Fig. 11** The confusion matrices of different classifiers (PM, M and OM indicates premature, mature and over-mature, respectively)**Table 5** Classification summary along with the confusion matrix for classification of different medicinal plant leaves

Leaf classes	Precision	Sensitivity/recall	FMeasure	Specificity
Kalmegh	1.00	1.00	0.99	0.98
Neem	0.99	0.98	0.98	
Tulsi	0.99	1.00	0.99	
Micro avg	1.00	0.99	0.99	
Macro avg	0.99	0.99	0.99	
Weighted avg	1.00	0.99	0.99	

**Table 6** Classification summary along with the confusion matrix for classification of the maturity grades of different medicinal plant leaves

Leaf types	Maturity type	Precision	Sensitivity/recall	FMeasure	Specificity
Kalmegh	Premature	0.99	1.00	1.00	0.98
	Mature	0.99	0.98	0.98	
	Over-mature	0.99	1.00	0.99	
	Micro-avg	0.99	0.99	0.99	
	Macro-avg	0.99	0.98	0.99	
	Weighted avg	1.00	0.99	1.00	
Neem	Premature	1.00	0.99	0.99	0.99
	Mature	1.00	0.99	0.99	
	Over-mature	0.98	0.99	0.98	
	Micro-avg	0.99	0.99	0.99	
	Macro-avg	0.99	0.99	0.99	
	Weighted avg	0.99	0.99	0.99	
Tulsi	Premature	0.99	0.98	0.99	0.99
	Mature	1.00	0.99	0.99	
	Over-mature	0.99	1.00	1.00	
	Micro-avg	0.99	0.99	0.99	
	Macro-avg	0.99	0.99	0.99	
	Weighted avg	1.00	1.00	0.99	

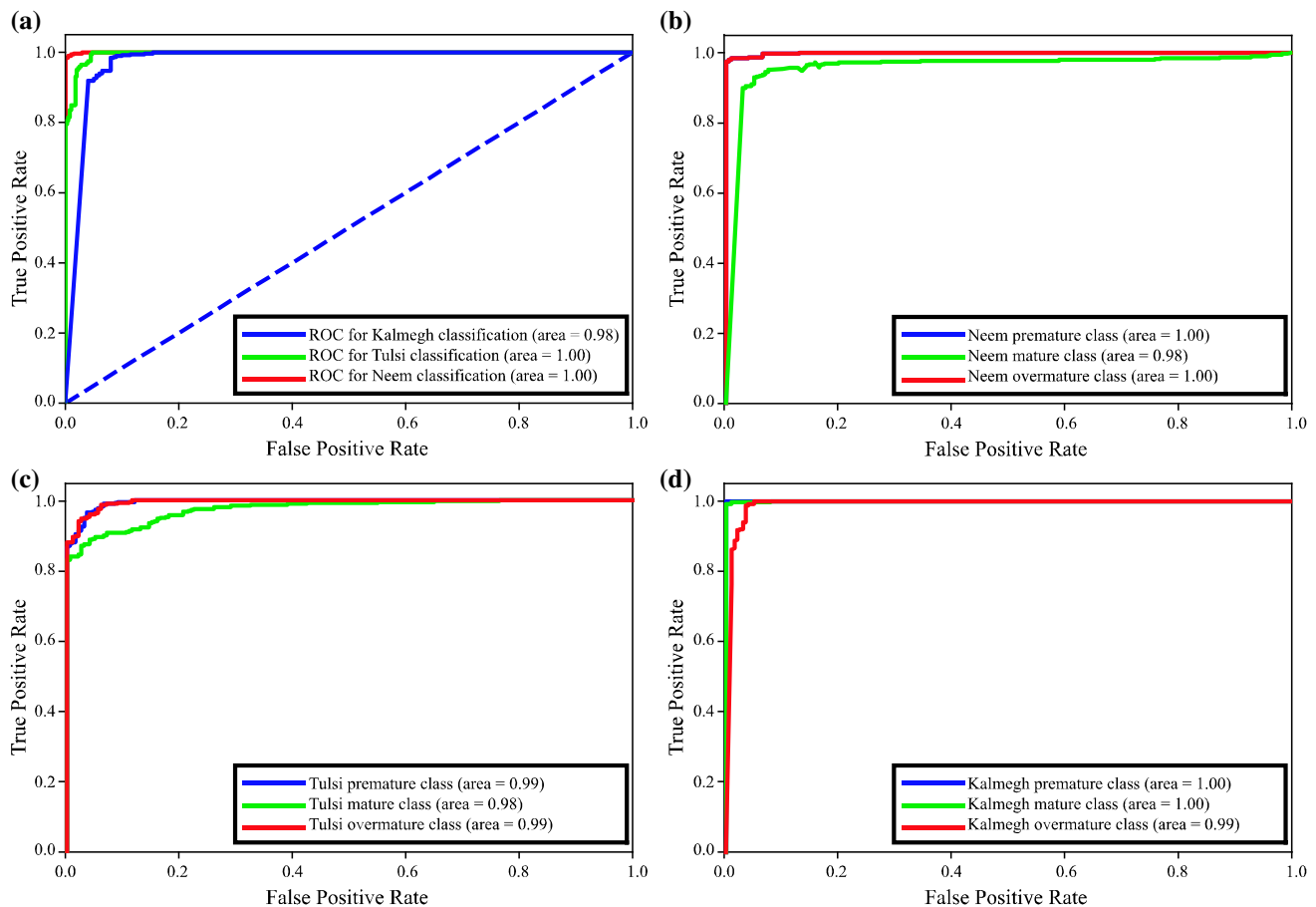
**Table 7** Cohen Kappa scores for classification models

Classifier	Classes taken for measurement	Cohen Kappa score
Leaf species classifier	Kalmegh, Neem and Tulsi leaves	0.9988
	Pre-mature, mature and over-mature	
Maturity stage classifier	Kalmegh	0.9998
	Neem	0.9988
	Tulsi	0.9980

was performed where the statistical moments of different color channels from three color spaces, namely, RGB, HSV and Lab, were considered as featureset. As shown in Table 8, the prediction results of presented system come in the form of a vector with classes marked as 1 are the final decisions of classifier.

The correct correspondence between the prediction vector value and the input leaf along with correct prediction of maturity stage confirms the high precision and accuracy of the presented method. Table 9 shows that CNN-based model performs visibly better than other techniques under consideration. It also conveys the improved performance of CNN models for both species classification and maturity stage classification purposes. The higher overall accuracy conveys the ability of the

presented model toward correct identification of leaf species as well as the maturity stage of the leaves, while higher FMeasure values represents a better balance between precision and recall of the presented model which in turn conveys the correctness of leaf identification in terms of its correct species and maturity stage. It can also be noted that about 99% accuracy of the presented model makes it competitive to the existing computer vision-based methods of leaf species identification (as stated in introduction section) where the accuracy range is 94–99% and as well as for medicinal leaves where the accuracy range is 85–91%. Hence, in light of the performance metrics the presented CNN model can be considered as a potential addition to the existing methods for the subjected problem.















**Fig. 12** ROC of different CNN classifiers **a** leaf species classifier, maturity state classifier of **b** Neem, **c** Tulsi and **d** Kalmegh leaves

## 4 Conclusion

The paper presented a computer vision-based approach for simultaneous classification of medicinal leaf species and corresponding maturity level. A CNN-based architecture has been explored with four CNN models trained with preprocessed images. The case studies with three species of medicinal leaves, namely, Neem, Tulsi and Kalmegh, have been presented. The performance of the model has been evaluated against different popular metrics. The presented

framework shows about 99% classification accuracy in terms of both identification of medicinal leaf species and the corresponding maturity level. The study can be further extended to exploring potential of other deep learning based classifiers and other soft computing approaches for hyperparameter optimization. Extension of the presented method to other medicinal plant leaves can play a pivotal role to identify the maturity while using them for medicinal purposes. The promising performance of the model motivates toward its extended applications in different

**Table 8** Presentation of the prediction vector and the predicted result for any leaf of unknown class and unknown maturity grade

Categories of classes	Image of sample	Prediction vector	Predicted result
Leaf species classifier		[ 1, 0, 0]	Kalmegh
		[0, 1, 0]	Neem
		[0, 0, 1]	Tulsi
Maturity stage classifiers			
Kalmegh		[1, 0, 0]	Premature
		[0, 1, 0]	Mature
		[0, 0, 1]	Overmature
Neem		[1, 0, 0]	Premature
		[0, 1, 0]	Mature
		[0, 0, 1]	Overmature
Tulsi		[1, 0, 0]	Premature
		[0, 1, 0]	Mature
		[0, 0, 1]	Overmature



**Table 9** The average performance measure of the confusion matrix estimated for the maturity detection by BPMLP and SVM methods

Method	Species classification/Maturity stage classification				
	Accuracy	Sensitivity	Specificity	Precision	FMeasure
BP MLP	0.97/0.96	0.98/0.97	0.96/0.96	0.97/0.97	0.96/0.96
SVM	0.96/0.96	0.95/0.95	0.96/0.96	0.96/0.96	0.95/0.95
Presented CNN model	0.99/0.99	0.99/0.99	0.98/0.99	1.00/0.99	0.99/0.99

problems. The work can further be extended to a mobile app development and a handheld mobile device development which may be helpful for faster gradation in real-time applications.

**Funding** The authors have no relevant financial or non-financial interests to disclose.

**Data availability** The datasets generated during and/or analyzed during the current study are not publicly available due to internal research policy but are available from the corresponding author on reasonable request.

## Declarations

**Conflict of interest** Authors declare no conflicts of interest/competing interests.

## References

- Abadi M et al. (2016) TensorFlow: A system for large-scale machine learning. In: 12th USENIX Symposium on operating systems design and implementation (OSDI 16), USENIX Association: 265–283.
- Ahmed B et al (2021) Experimental investigation, binary modelling and artificial neural network prediction of surfactant adsorption for enhanced oil recovery application. *Chem Eng J* 406:127081. <https://doi.org/10.1016/j.cej.2020.127081>
- Alzohairy M (2016) Therapeutics role of *Azadirachta indica* (Neem) and their active constituents in diseases prevention and treatment. *Evidence-Based Complement Altern Med*. <https://doi.org/10.1155/2016/7382506>
- Attanayake R et al (2019) The effect of maturity status on biochemical composition, antioxidant activity and anthocyanin biosynthesis gene expression in a pomegranate (*Punica granatum* L.) cultivars with red flowers, yellow peel, and pinkish arils. *J Plant Growth Regul* 38(3):992–1006
- Berrar D (2019) Performance measures for binary classification. In: Ranganathan S, Gribskov M, Nakai K, Schönbach C (eds) *Encyclopedia of bioinformatics and computational biology*. Academic Press, Oxford, pp 546–560. <https://doi.org/10.1016/B978-0-12-809633-8.20351-8>
- Bhuiyan MR et al (2021) MediNET: A deep learning approach to recognize Bangladeshi ordinary medicinal plants using CNN. In: Borah S, Pradhan R, Dey N, Gupta P (eds) *Soft computing techniques and applications: advances in intelligent systems and computing*. Springer, Singapore
- Bojovic B, Stojanovic J (2005) Chlorophyll and carotenoid content in wheat cultivars as a function of mineral nutrition. *Arch Biol Sci, Belgrade* 57(4):283–290
- Buitinck L et al. (2013) API design for machine learning software: experiences from the scikit-learn project. In: *European conference on machine learning and principles and practices of knowledge discovery in databases*. [arXiv:1309.0238](https://arxiv.org/abs/1309.0238)
- Catur P, Mohammad D, Hasta M (2020) Implementation of CNN for plant leaf classification. *Int J Inform Comput*. <https://doi.org/10.35842/ijicom.v2i2.28>
- Chaki J, Parekh R, Bhattacharya S (2015) Recognition of whole and deformed plant leaves using statistical shape features and neuro-fuzzy classifier. In: *IEEE Proceedings of 2nd international conference on recent trends in information system (ReTIS): 2015*
- Chollet F (2015) Keras. <https://github.com/fchollet/keras>, Accessed on 10 January 2020
- Costa N, Lima M, Rommel B (2020) Evaluation of feature selection methods based on artificial neural network weights. *Expert Syst Appl*. <https://doi.org/10.1016/j.eswa.2020.114312>
- Deepalakshmi P, Lavanya K, Srinivasu PN (2021) Plant leaf disease detection using CNN algorithm. *Int J Inf Syst Model Des (IJISMD)* 12(1):1–21
- Eid HF, Abraham A (2018) Plant species identification using leaf biometrics and swarm optimization: a hybrid PSO GWO, SVM Model. *Int J Hybrid Intell Syst* 14(2):1–11
- Faridi H, Aboonajmi M (2017) Application of machine vision in agricultural products. In: *Proceedings 4th Iranian international NDT conference*, Olympic Hotel, Tehran, Iran, Feb 26–27
- Fawcett T (2006) An introduction to ROC analysis. *Pattern Recogn Lett* 27:861–874
- Hinton E, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief networks. *Neural Comput* 18(7):1527–1554
- Jaswal D, Sowmya V, Soman KP (2014) Image classification using convolutional neural networks. *Int J Adv Res Technol* 3(6):1661–1668
- Jeon WS, Rhee SY (2017) Plant leaf recognition using a convolution neural network. *Int J Fuzzy Log Intell Syst* 17:26–34
- Kamble PN et al (2015) Estimation of Chlorophyll content in young and adult leaves of some selected plants. *Univers J Environ Res Technol* 5(6):306–310
- Karthik R, Hariharan M, Anand S et al (2019) Attention embedded residual CNN for disease detection in tomato leaves. *Appl Soft Comput*. <https://doi.org/10.1016/j.asoc.2019.105933>
- Karthik R, Hariharan M, Anand S et al (2020) Attention embedded residual CNN for disease detection in tomato leaves. *Appl Soft Comput* 86:105933
- Keivani M, Mazloun J, Sedaghatfar E, Tavakoli MB (2020) Automated analysis of leaf shape, texture, and color features for plant classification. *Traitement du Signal* 37(1):17–28. <https://doi.org/10.18280/ts.370103>
- Kumkar S, Dobos GJ, Ramp T (2017) The significance of ayurvedic medicinal plants. *J Evid Based Complement Altern Med* 22(3):494–501
- LeCun Y, Bengio Y, Hinton GR (2015) Deep learning. *Nature* 521(7553):436–444
- Li Y et al (2021) Towards a comprehensive optimization of engine efficiency and emissions by coupling artificial neural network (ANN) with genetic algorithm (GA). *Energy* 225:120331. <https://doi.org/10.1016/j.energy.2021.120331>

- Lipton ZC, Berkowitz J (2015) A critical review of recurrent neural networks for sequence learning, [arXiv:1506.00019v4](https://arxiv.org/abs/1506.00019v4) [cs.LG] 17 Oct, 2015.
- Mahajan S, Raina A, Gao X-Z, Pandit K (2021) A plant recognition using morphological feature extraction and transfer learning over SVM and AdaBoost. *Symmetry* 13:356. <https://doi.org/10.3390/sym13020356>
- McHugh ML (2012) Interrater reliability: the kappa statistic. *Biochem Med* 22(3):276–282
- Panigrahi K, Sahoo A, Das H (2020). A CNN approach for corn leaves disease detection to support digital agricultural system. pp. 678–683. <https://doi.org/10.1109/ICOEI48184.2020.9142871>.
- Rafiq A et al (2013) Application of computer vision system in food processing- A review. *J Eng Res Appl* 3(6):1197–1205
- Renuka B, Sanjeev B, Ranganathan D (2016) Evaluation of phyto-constituents of *Caralluma Nilagiriana* by FTIR and UV-VIS spectroscopic analysis. *J Pharmacogn Phytochem* 5(2):105–108
- Rere LMR, Fanany MI, Arymurthy AM (2016) Metaheuristic algorithms for convolution neural network. *Comput Intell Neurosci*. <https://doi.org/10.1155/2016/1537325>
- Salle A, Villavicencio A (2018) Restricted recurrent neural Tensor networks: Exploiting word frequency and compositionality. In: *Proceedings of the 56th annual meeting of the association for computational linguistics (Short Papers)*: 8–13, Melbourne, Australia, July 15 – 20
- Santra AK, Christy CJ (2012) Genetic algorithm and confusion matrix for document clustering. *Int J Comput Sci* 9(1):322–328
- Sapijaszko G, Mikhael WB (2018) An overview of recent convolutional neural network algorithms for image recognition. In: *2018 IEEE 61st International midwest symposium on circuits and systems (MWSCAS)*, 2018, pp. 743–746. <https://doi.org/10.1109/MWSCAS.2018.8623911>
- Sardogan M, Tuncer A, Ozen Y (2018, September). Plant leaf disease detection and classification based on CNN with LVQ algorithm. In: *2018 3rd International conference on computer science and engineering (UBMK)* (pp. 382–385). IEEE
- Traorea BB, Foguea BK, Tangara F (2018) Deep convolution neural network for image recognition. *Eco Inform* 48:257–268
- Turkoglu M, Hanbay D (2019) Recognition of plant leaves: an approach with hybrid features produced by dividing leaf images into two and four parts. *Appl Math Comput* 352:1–14
- Upadhyay RK (2017) Tulsi: a holy plant with high medicinal and therapeutic value. *Int J Green Pharm (Suppl)* 11(1):S1–S12
- Venkataraman D, Mangayarkarasi N (2016) Computer vision based feature extraction of leaves for identification of medicinal values of plants. In: *IEEE International conference on computational intelligence and computing research*: 978–1–5090–0612–0/16
- Verma H et al (2019) Evaluation of an emerging medicinal crop Kalmegh [*Andrographis paniculata* (Burm. F.) Wall Ex. Nees] for commercial cultivation and pharmaceutical & industrial uses: a review. *J Pharmacogn Phytochem* 8(4):835–838
- Wang G et al (2019) A PSO and BFO-based learning strategy applied to faster R-CNN for object detection in autonomous driving. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2019.2897283>
- Wang Y, Zhang H, Zhang G (2019b) cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks. *Swarm Evol Comput* 49:114–123
- Wilfa P et al (2016) Computer vision cracks the leaf code. *PNAS* 113(12):3305–3310
- Yamashita R et al (2018) Convolutional neural networks: an overview and application in radiology. *Insights Imaging* 9:611–629
- Yuanita AP, Esmeralda CD, Ridwan I (2021) Identification of medicinal plant leaves using convolutional neural network. *J Phys: Conf Ser* 1845:012026. <https://doi.org/10.1088/1742-6596/1845/1/012026>

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

## Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).

Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.

These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.

We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.

While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval, sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.

These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.

To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.

Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.

If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

[onlineservice@springernature.com](mailto:onlineservice@springernature.com)