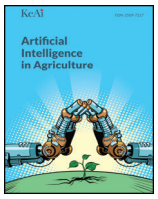
Artificial Intelligence in Agriculture 5 (2021) 90–101

Contents lists available at ScienceDirect

Artificial Intelligence in Agriculture

journal homepage: http://www.keaipublishing.com/en/journals/artificial-intelligence-in-agriculture/



Plant disease detection using hybrid model based on convolutional autoencoder and convolutional neural network

Punam Bedi, Pushkar Gole *

Department of Computer Science, University of Delhi, Delhi, India

ARTICLE INFO

Article history:
Received 29 September 2020
Received in revised form 7 May 2021
Accepted 8 May 2021
Available online 11 May 2021

Keywords:
Plant disease detection
Convolutional autoencoder
Convolutional neural network
Deep learning in agriculture

ABSTRACT

Plants are susceptive to various diseases in their growing phases. Early detection of diseases in plants is one of the most challenging problems in agriculture. If the diseases are not identified in the early stages, then they may adversely affect the total yield, resulting in a decrease in the farmers' profits. To overcome this problem, many researchers have presented different state-of-the-art systems based on Deep Learning and Machine Learning approaches. However, most of these systems either use millions of training parameters or have low classification accuracies. This paper proposes a novel hybrid model based on Convolutional Autoencoder (CAE) network and Convolutional Neural Network (CNN) for automatic plant disease detection. To the best of our knowledge, a hybrid system based on CAE and CNN to detect plant diseases automatically has not been proposed in any state-of-the-art systems present in the literature. In this work, the proposed hybrid model is applied to detect Bacterial Spot disease present in peach plants using their leaf images, however, it can be used for any plant disease detection. The experiments performed in this paper use a publicly available dataset named PlantVillage to get the leaf images of peach plants. The proposed system achieves 99.35% training accuracy and 98.38% testing accuracy using only 9,914 training parameters. The proposed hybrid model requires lesser number of training parameters as compared to other approaches existing in the literature. This, in turn, significantly decreases the time required to train the model for automatic plant disease detection and the time required to identify the disease in plants using the trained model.

1. Introduction

India is an agrarian country, and a major part of its economy depends on the agricultural sector. The share of agriculture in the Indian Gross Domestic Product (GDP) and total exports are 16% and 10%, respectively (Himani, 2014). About 75% population of India depends on the agricultural sector either directly or indirectly (Himani, 2014). Therefore, disease-free good quality crop production is essential for the growth of the country's economy.

Like human beings, plants are also susceptible to various kinds of diseases in their different phenophases. Consequently, the total crop yield and hence the net profit of the farmer are adversely affected. In order to address this issue, the early detection of plant diseases is necessary. Manual disease detection in plants is done either by farmers or by agricultural scientists. However, this is a very challenging and time-consuming task. To address this problem, many researchers across the globe presented different state-of-the-art systems for automatic plant disease detection with the help of various Machine Learning (Ahmed et al., 2019; Naik and Sivappagari, 2016; Panigrahi Kshyanaprava Panda et al., 2020; RAO et al., 2020), and Deep Learning techniques (Ashraf and Khan, 2020; Chen et al., 2020; Karlekar and Seal, 2020; Kim et al., 2020). These state-of-the-art systems use a very high number of training parameters. Consequently, the training time and the prediction time of these systems are very high, or they require a machine with high computation powers. This research work tries to reduce the number of features used for prediction using the CAE network without a significant decrease in the classification accuracy of plant disease detection. This, in turn, decreases the number of training parameters by a significant factor resulting in the decrease of training and prediction time.

Deep Learning techniques are inspired by the architecture of neurons present in the human brain (Haykin, 1998). These techniques use Artificial Neural Networks (ANNs), and its other variants, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to identify the hidden structures in data. There are two prominent advantages of Deep Learning techniques over the Machine Learning techniques. First, they automatically extract various features from raw data, and hence there is no need for an extra feature extraction module. Second, Deep Learning techniques reduce the amount of time required to process large datasets of high dimensions. Therefore, the Deep Learning techniques are used to build the proposed hybrid model.

* Corresponding author.
E-mail addresses: pbedi@cs.du.ac.in (P. Bedi), pgole@cs.du.ac.in (P. Gole).

https://doi.org/10.1016/j.aiia.2021.05.002

Table 1
Summary of various research work.

Author and year	Dataset used	Best model	Testing accuracy	Number of training parameters
Sanga et al. (2020)	Banana leaf images obtained from banana field	ResNet-152	99.2%	60 million
Chohan et al. (2020)	PlantVillage	VGG-19	98.3%	143 million
Ferentinos (2018)	PlantVillage	VGGNet	99.5%	138 million
Mohanty et al. (2016)	PlantVillage	GoogLeNet	99.3%	7 million
Mohameth et al. (2020)	PlantVillage	ResNet-50 + SVM	98%	25 million
Tiwari et al. (2020)	Potato leaf extracted from PlantVillage dataset	VGG-19 + Logistic Regression	97.8%	143 million
Khamparia et al. (2020)	Tomato, potato, and maize leaf extracted from PlantVillage	CAE	86.78%	3.3 million

Convolutional Neural Networks (CNNs) and Convolutional Autoencoders (CAEs) are two Deep Learning techniques used in many computer vision applications due to their effectiveness on image data. Both these techniques use convolution operation to extract various spatial and temporal features from image data. CNNs are used to classify input images to their respective classes, whereas CAEs are used to reduce the dimensionality of an image efficiently.

This paper proposes a novel hybrid model for automatic plant disease detection based on CAE and CNN with fewer training parameters as compared to other state-of-the-art systems present in the literature. Although there are various techniques present in the literature used for automatic plant disease detection, but a hybrid system of CAE combined with a CNN has not been proposed until now in any existing research work to the best of our knowledge. Dimensionality reduction using CAE in the proposed model results in reduction of number of training parameters of the model. In order to test the proposed hybrid model, it is applied to detect Bacterial Spot disease in peach plant, which is caused by a bacterium named Xanthomonas Campestris. This model can be used to detect other plant diseases, as well.

The rest of the paper is divided into four sections. Section 2 discuss several state-of-the-art systems for automatic plant disease detection present in the literature. In Section 3, the material and methods used to design proposed hybrid model is described. The results obtained by the model to detect Bacterial Spot disease in peach plants are presented in Section 4 of the paper. In the end, Section 5 concludes the paper.

2. Related work

Many researchers across the world explored different Machine Learning (Es-saady et al., 2016; Islam et al., 2017; Krithika and Selvarani, 2017; Padol and Yadav, 2016), Deep Learning (Golhani et al., 2018; Ramesh and Vydeki, 2020; Sharma et al., 2019), Image Processing (Marwaha et al., 2012; Ngugi et al., 2020; Qin et al., 2016; Tewari et al., 2020), Soft Computing (Singh, 2019; Singh and Misra, 2017), and Semantic Web-based (Jearanaiwongkul et al., 2018; Marwaha et al., 2009) techniques to automate plant disease detection. This section discusses some state-of-the-art systems present in the literature used for automatic plant disease detection.

Sanga et al. (Sanga et al., 2020) developed a disease detection tool for banana plants with five different CNN architectures. These architectures were VGG-16, ResNet-152, ResNet-50, ResNet-18, and InceptionV3. They found that ResNet-152 outperformed others with an accuracy of 99.2%. They also developed a mobile application so that farmers could easily detect diseases in banana plants by uploading leaf images of their banana plants with their smartphones. This mobile application used the InceptionV3 model for disease prediction with 99% confidence. The number of training parameters used by their best performing model, i.e., ResNet-152, was 60 million, as mentioned in the ResNet paper (He et al., 2016). Another similar work was done in the paper authored by Chohan et al. (Chohan et al., 2020). They employed VGG-19 and InceptionV3 CNN architectures for automatic plant disease detection using the PlantVillage dataset. In their research work, they also used data augmentation to enlarge the dataset artificially. The VGG-19 model outperformed the InceptionV3 model with 98% training accuracy and 95% testing accuracy, as claimed by them in their paper. The number of training parameters used by their best performing model, i.e., VGG-19, was 143 million, as claimed by (Simonyan and Zisserman, 2015) in their research work.

Ferentinos (Ferentinos, 2018) employed five different modern CNN architectures named AlexNet, AlexNetOWTBn, GoogLeNet, Overfeat, and VGG for plant disease detection using the PlantVillage dataset. In his paper, he found that VGG outperformed other CNN architectures with an accuracy of 99.5% using 138 million trainable parameters, as mentioned in the VGGNet paper (Simonyan and Zisserman, 2015). Mohanty et al. (Mohanty et al., 2016) analyzed AlexNet and GoogLeNet CNN architectures' performance for plant disease detection using the PlantVillage dataset. They performed 60 different experiments with the help of 60 different configurations. They found that GoogLeNet with transfer learning performed best with an accuracy of 99.3%. The number of training parameters used by GoogLeNet was around 7 million, as claimed in their paper (Szegedy et al., 2015).

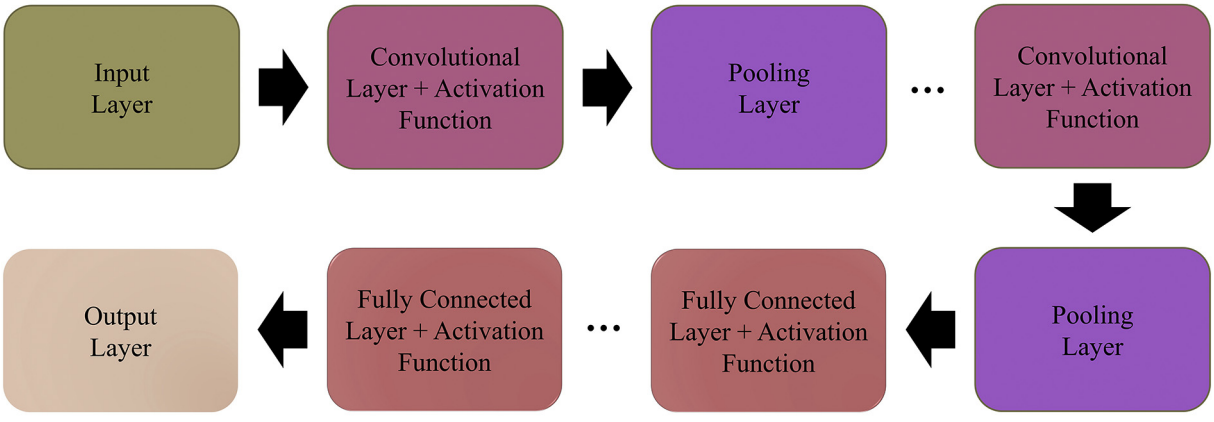


Fig. 1. The architecture of a typical CNN.

Mohameth et al. (Mohameth et al., 2020) used different modern CNN architectures and different classifiers for automatic plant disease detection on the PlantVillage dataset. They employed VGG-16, ResNet-50, and GoogLeNet CNN architectures for feature extraction, and for classification, they used k-Nearest Neighbour and Support Vector Machine (SVM) classifiers. They observed that SVM with ResNet-50 outperformed others with an accuracy of 98%. As mentioned in the ResNet paper, the number of training parameters used by ResNet-50 was approximately 25 million (He et al., 2016). Similar work was also done by Tiwari et al. (Tiwari et al., 2020). They proposed an automatic disease detection system for potato plants. This system used different CNN architectures such as VGG-19, VGG-16, and InceptionV3 for feature extraction and different classifiers such as Logistic Regression, k-Nearest Neighbour classifier, Support Vector Machine (SVM), and Neural Network for disease detection. They concluded that VGG-19 with Logistic Regression outperformed others with an accuracy of 97.8%. The number of training parameters used by VGG-19 was approximately 143 million, as claimed in the VGGNet paper (Simonyan and Zisserman, 2015).

Khamparia et al. (Khamparia et al., 2020) proposed a Deep Convolutional Encoder Network system for seasonal crops disease identification. They considered 900 leaf images of three crops: potato, tomato, and maize, distributed in six classes (i.e., five diseased and one healthy). They achieved 100% training accuracy while the testing accuracy of their model was 86.78%. Since the training accuracy was much higher as compared to the testing accuracy, so there was a chance that the trained model overfitted on the training data. In their paper, they have also mentioned that their system used approximately 3.3 million training parameters, which is much higher than the number of training parameters 9,914 used in the proposed work. They have used Autoencoder and CNN for seasonal crop disease identification. On the other hand, the proposed novel hybrid model is based on CAE and CNN. Moreover, the proposed model also achieves higher testing accuracy than the testing accuracy of the model proposed by Khamparia et al. (Khamparia et al., 2020). Pardede et al. (Pardede et al., 2018) designed a system for automatic disease detection for corn and potato plants with CAE and SVM classifiers' help. They extracted leaf images of potato and corn plants from the PlantVillage dataset. They achieved 87.01% and 80.42% accuracy in detecting diseases in potato and corn plants, respectively.

As per our knowledge, state-of-the-art systems available in the literature for plant disease detection use a very high number of training parameters ranging from 3.3 million to 143 million (He et al., 2016; Khamparia et al., 2020; Simonyan and Zisserman, 2015; Szegedy et al., 2015). A summary of these research works has been presented in Table 1.

All of the above-discussed research works have a major disadvantage that all research works used a very high number of training parameters. Moreover, training a model with a very high number of training parameters requires either a lot of training time or a machine with high computation power. This motivated us to work towards reducing the number of training parameters used for plant disease detection without much decrease in the classification accuracy. Hence in this paper, a novel hybrid model is proposed that reduces the dimensionality of input leaf image using CAE before classifying it using CNN for plant disease detection. The dimensionality reduction of plant leaf images before classification reduces the number of training parameters by a significant factor, which is the major finding of this research work.

3. Material and methods

This section discusses the material and methods which are used to design the model. Section 3.1 provides a basic understanding of CNN and CAE, which is helpful to understand the proposed work. In Section 3.2, the proposed hybrid model is described in detail. The experimental configuration to implement the proposed model is presented in Section 3.3.

3.1. Background concepts

This section describes the basic concepts of Convolutional Neural Network (CNN) and Convolutional Autoencoder (CAE) which are used to design the proposed hybrid model.

3.1.1. Convolutional neural network (CNN)

Convolutional Neural Network is a Deep Learning technique that uses convolution operation instead of simple matrix multiplication. As compared to other Deep Learning techniques, CNN deals with images most efficiently. It extracts different spatial and temporal features

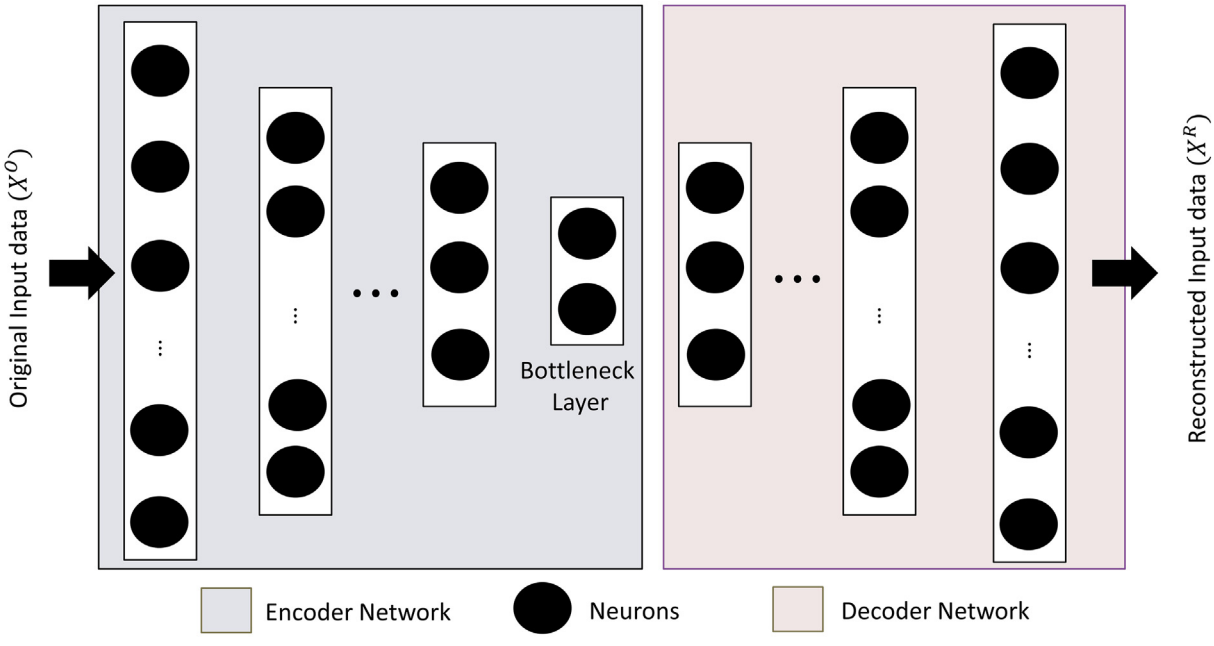


Fig. 2. Architecture of a N layer Autoencoder.

from input images, which play a significant role in image classification and other computer vision tasks. Fig. 1 depicts the architecture of a typical CNN that contains one Input layer, one Output layer, a set of Convolutional layers (each with an activation function), Pooling layers, and Fully Connected layers (each with an activation function).

The Convolutional layer present in the CNN performs the convolution operation. The initial Convolutional layers of a CNN extract the simple lower-level features of an image, and the Convolutional layers present at the end of the network extract the complex higher-level features of an image. The convolution operation is defined as a binary operation (represented by symbol '∗') between two real-valued functions (say $f(x)$ and $g(x)$). In the continuous domain, it can be mathematically defined as in Eq. (1). Similarly, in the discrete domain, the mathematical formula for convolution operation can be written in Eq. (2) (Goodfellow et al., 2016).

$$f(x)*g(x) = \int f(x) \cdot g(x-k)dk \tag{1}$$

$$f(x)*g(x) = \sum_{k=-\infty}^{\infty} f(x) \cdot g(x-k) \tag{2}$$

In CNN, $f(x)$ and $g(x)$ are termed, as input and filter/kernel, respectively, and the output of the convolution operation is known as the feature map. The input, kernel/ filter, and feature map are stored as multidimensional arrays. From the definition of convolution operation, it can be observed that if the size of the input matrix is $m \times m$ and the size of the filter is $k \times k$ (where $k \leq m$), then the size of the output feature map is $m-k+1 \times m-k+1$. Thus, it can be concluded that after each convolution operation, the size of output feature map is decreased. In other words, the size of the input image reduces after each convolution operation and becomes zero after some convolutions. Hence, it limits CNN's depth by placing an upper bound on the number of Convolutional layers present in a CNN. Further, the elements present on the edges and corners are used less than the elements present in the center of the input matrix. To tackle these two issues, padding is used in the Convolutional layers present in the CNN.

Padding is used to expand the input matrix by appending the layers of zeroes to the input matrix's border. Thus, the input matrix area is increased on which the convolution operation has to be performed, which ensures that the size of the input matrix does not decrease after convolution operation. It also ensures that the elements present on the edges and corners are also utilized by adding multiple padding layers. There are two types of padding: Valid Padding and Same Padding. In Valid Padding, no layers of zeroes are appended to the input matrix, and hence the dimensionality of the output matrix remains the same as illustrated above, i.e., $m-k+1 \times m-k+1$. On the other hand, in the Same Padding, *p layers* of zeroes are appended to the input matrix such that its dimensionality does not change after the convolution operation. The value of p can be determined by Eq. (3). Same Padding has been used to design the proposed hybrid model.

$$m+2p-k+1 = m \Rightarrow p = \frac{k-1}{2} \tag{3}$$

From Eqs. (1) and (2), it can be observed that convolution is a linear operation. Therefore, to extract the non-linear features from images, different non-linear activation functions such as Sigmoid, Hyperbolic Tangent, Rectified Linear Unit (ReLu), etc., are used in CNNs after the convolution operation.

After applying non-linear activation functions, pooling operation is performed. Pooling operation is used to reduce the number of training parameters and hence reduce the dimensionality of the feature map it receives from its preceding layer. It computes a single output value based on some statistics from its neighborhood. Some of these statistics are Max Pooling, Average Pooling, etc. Max Pooling picks the maximum value from its neighborhood, and Average Pooling computes the average value in its neighborhood.

3.1.2. Convolutional autoencoder (CAE)

Autoencoder is a self-supervised learning algorithm that uses a Neural Network for representation learning (Bisong, 2019). Representation learning is a technique in which a system learns to encode input data. Autoencoders are used to map input data to some lower-dimensional space or compressed domain representation. To do this, a bottleneck is introduced in the network, which enforces the system to learn the compressed domain representation of input data. Fig. 2 shows the architecture of a N layer Autoencoder.

An Autoencoder comprises four components: Encoder Network, Bottleneck Layer, Decoder Network, and Reconstruction Loss (Goodfellow et al., 2016). Encoder Network is a Neural Network that encodes input data to a compressed domain. The Bottleneck layer is the last layer of the Encoder Network, and its output is known as encoded input data. Let there are N layers in an Encoder Network in which the last layer i.e., N^{th} layer is Bottleneck Layer. The mathematical equation to represent the working of each layer present in the Encoder Network is shown in Eq. (4).

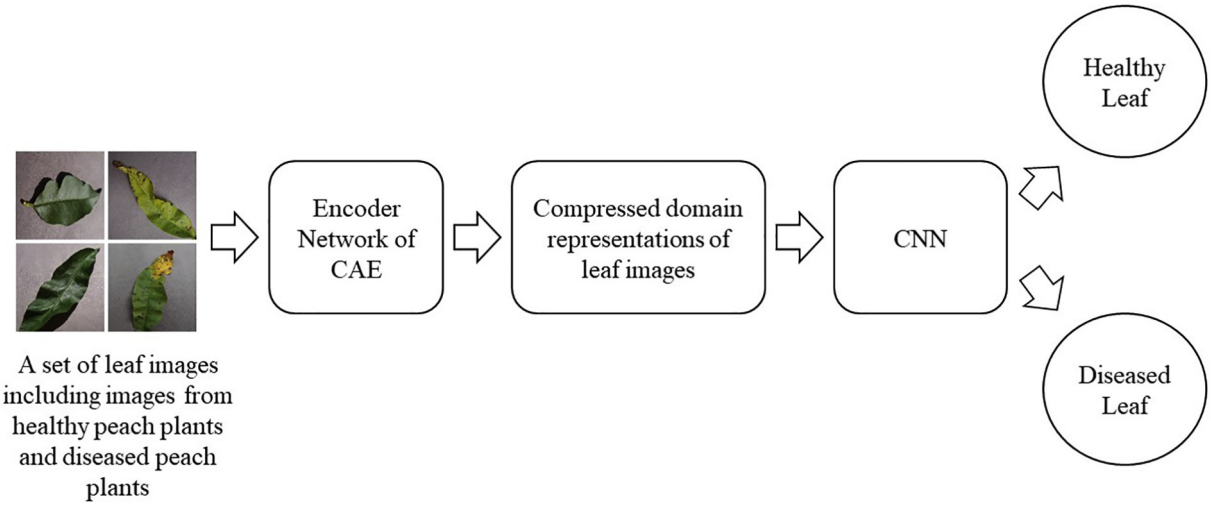


Fig. 3. Block diagram of the proposed hybrid model.

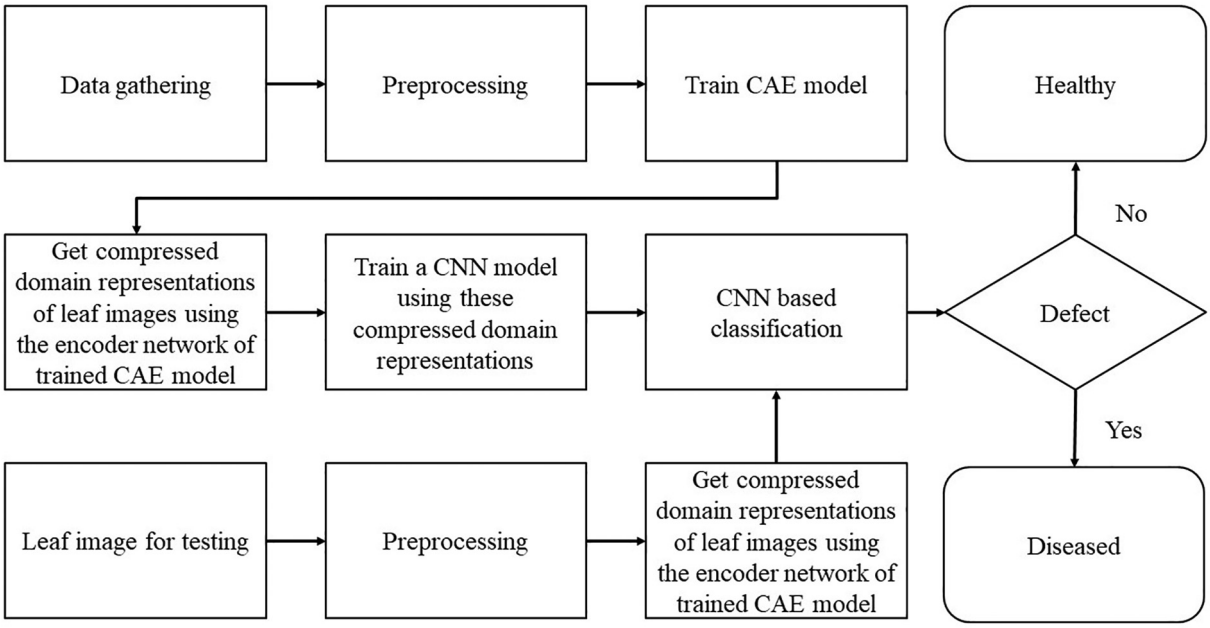


Fig. 4. Flow-diagram to demonstrate the proposed methodology.

$$X_{ei+1} = f_{e_i}\left(W_{e_i}^T X_{e_i} + b_{e_i}\right) \forall i = 0, 1, 2, \ldots, N \quad (4)$$

where X_{ei} is the input for the i^{th} layer of Encoder Network, $X_{ei+1}$is the output of the i^{th} layer of the Encoder Network, W_{ei} is the weight vector for the i^{th} layer of the Encoder Network, b_{ei} is the bias for the i^{th} layer of the Encoder Network, and $f_{ei}$is the activation function for the i^{th} layer of the Encoder Network.

Decoder Network is also a Neural Network that takes the output of the Bottleneck Layer as input and tries to reconstruct the original data. The number of layers in the Decoder Network is the same as the number of layers in the Encoder Network but in the reverse order. The last layer of the Decoder Network produces the noisy reconstruction of input data. The mathematical equation to represent the working of each layer present in the Decoder Network is shown in Eq. (5).

$$X_{di+1} = f_{d_i}\left(W_{d_i}^T X_{d_i} + b_{d_i}\right) \forall i = 0, 1, 2, \cdots, N \quad (5)$$

where X_{di} is the input for the i^{th} layer of Decoder Network, $X_{di+1}$is the output of the i^{th} layer of the Decoder Network, W_{di} is the weight vector for the i^{th} layer of the Decoder Network, b_{di} is the bias for the i^{th} layer of the Decoder Network, and $f_{di}$is the activation function for the i^{th} layer of the Decoder Network. The difference between the original data X^O and the reconstructed data X^R is known as Reconstruction Loss. The autoencoder is trained using Backpropagation Algorithm to minimize the Reconstruction Loss. *Mean Squared Error (MSE)* and *Binary Cross-*

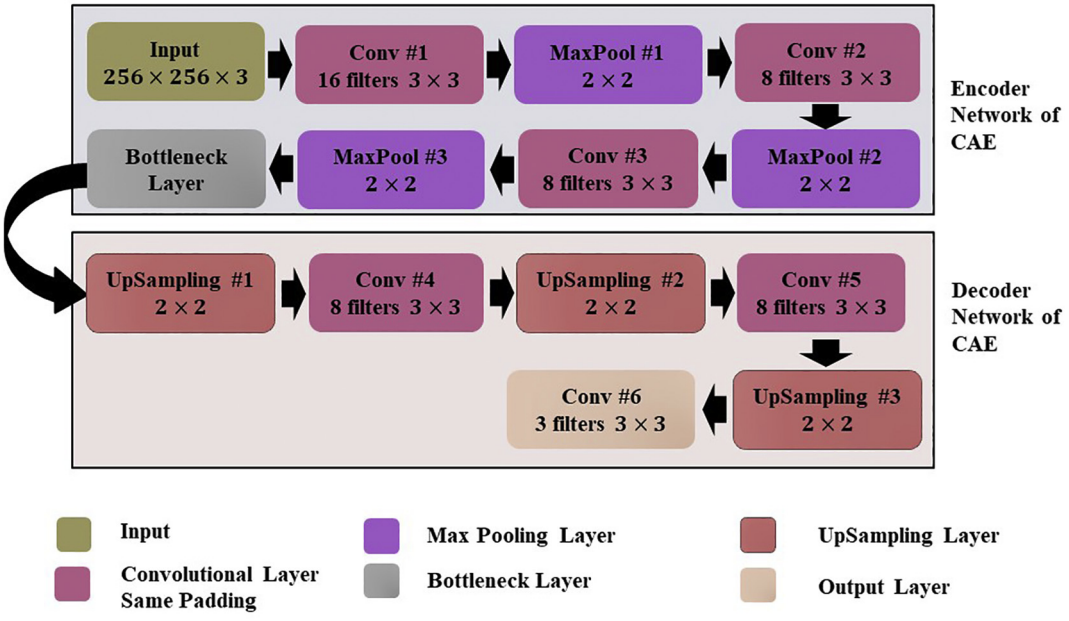


Fig. 5. CAE network architecture for the proposed hybrid model.

Table 2
Layer wise details of CAE network.

Layer number	Layer (type)	Input shape	Number of filters/channels	Size of each filter/channel	Activation function	Padding	Output shape	Number of training parameters
1	Input Layer	256 × 256 × 3	3	–	–	–	256 × 256 × 3	0
2	Conv #1	256 × 256 × 3	16	3 × 3	ReLu	Same Padding	256 × 256 × 16	448
3	MaxPool #1	256 × 256 × 16	16	2 × 2	–	–	128 × 128 × 16	0
4	Conv #2	128 × 128 × 16	8	3 × 3	ReLu	Same Padding	128 × 128 × 8	1,160
5	MaxPool #2	128 × 128 × 8	8	2 × 2	–	–	64 × 64 × 8	0
6	Conv #3	64 × 64 × 8	8	3 × 3	ReLu	Same Padding	64 × 64 × 8	584
7	MaxPool #3	64 × 64 × 8	8	2 × 2	–	–	32 × 32 × 8	0
8	Bottleneck Layer	32 × 32 × 8	8	3 × 3	ReLu	Same Padding	32 × 32 × 8	584
9	UpSampling Layer #1	32 × 32 × 8	8	2 × 2	–	–	64 × 64 × 8	0
10	Conv #4	64 × 64 × 8	8	3 × 3	ReLu	Same Padding	64 × 64 × 8	584
11	UpSampling Layer #2	64 × 64 × 8	8	2 × 2	–	–	128 × 128 × 8	0
12	Conv #5	128 × 128 × 8	8	3 × 3	ReLu	Same Padding	128 × 128 × 8	584
13	UpSampling Layer #3	128 × 128 × 8	8	2 × 2	–	–	256 × 256 × 8	0
14	Conv #6 (Output Layer)	256 × 256 × 8	3	3 × 3	ReLu	Same Padding	256 × 256 × 3	219
Total number of trainable parameters								**4,163**

Entropy (BCE) Loss are the two prominently used loss functions to compute Reconstruction Loss. The formulae for these two loss functions are shown in Eqs. (6) and (7), where D is the number of instances in a dataset on which the Autoencoder is applied.

$$MSE\left(X^O, X^R\right) = \frac{1}{D}\sum_{j=1}^{D}\left(X_j^O - X_j^R\right)^2 \tag{6}$$

$$BCE\left(X^O, X^R\right) = -\frac{1}{D}\sum_{j=1}^{D} X_j^O \cdot \log X_j^R + \left(1 - X_j^O\right) \cdot \log\left(1 - X_j^R\right) \tag{7}$$

There are different types of autoencoders, such as Undercomplete Autoencoder, Deep Autoencoder, Convolutional Autoencoder, etc. A CNN based autoencoder is known as Convolutional Autoencoder (CAE). It uses Convolutional and Down Sampling (Pooling) layers for encoding the input image to its compressed domain representation. Similarly, Up Sampling and Convolutional layers are used to reconstruct the original image using its compressed domain representation. Since this research work deals with plant leaf images, so CAE has been used to obtain the compressed domain representation before classification, in the proposed hybrid model. The model reduces the size of leaf images such that the prominent features of leaf images are not lost and are further used in classification. Due to compressed domain representations of leaf images, the number of features is reduced significantly, which eventually reduces the number of training parameters and reduces the time taken for the training of hybrid system and time taken for classification by the hybrid system.

3.2. Proposed work

In this research work, a novel hybrid model is designed to detect plant diseases automatically. To the best of our belief, a hybrid system for automatic plant disease detection based on CAE and CNN has not been proposed in any research work present in the literature. This model uses two deep learning techniques: CAE and CNN. First, the CAE network has been trained to reduce the dimensionality of the input leaf images. The dimensionality reduction of the leaf images has been done such that the important features of the leaf images are not lost. This has been ensured by applying the upper limit on Reconstruction Loss of CAE. After reducing the dimensionality of leaf images, the output of the Encoder Network of CAE, (i.e., compressed domain

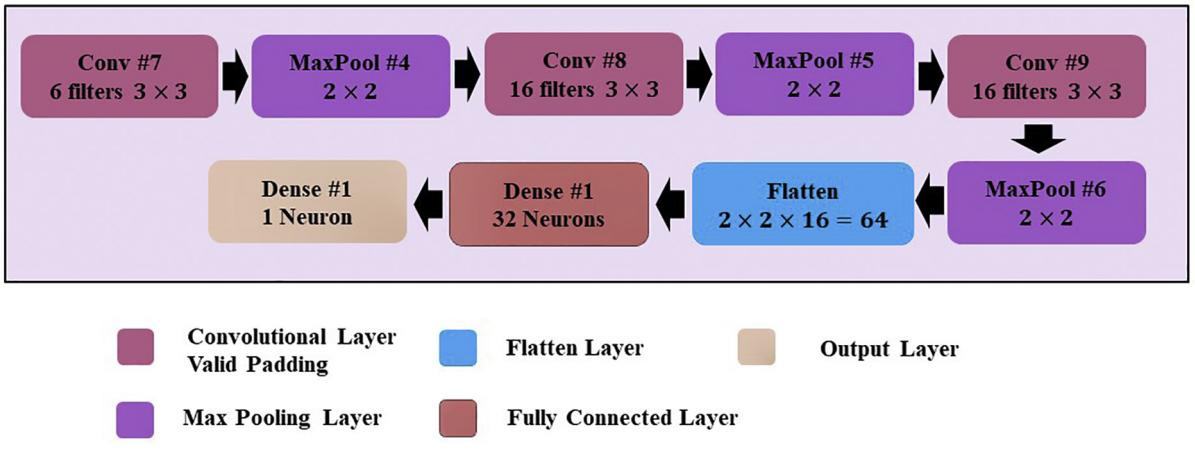


Fig. 6. CNN architecture for the proposed hybrid model.

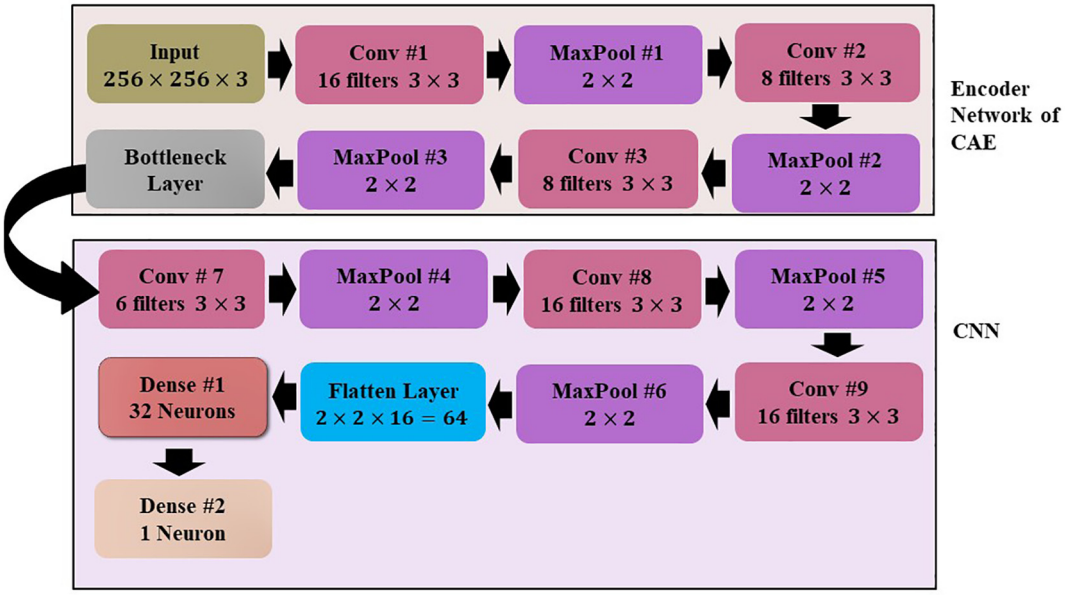


Fig. 7. Architecture of the proposed hybrid model.

representations of leaf images) is used as input to the CNN. With the help of CNN, the input leaf image has been classified as either a diseased leaf or a healthy leaf. The block diagram of the proposed hybrid model is shown in Fig. 3. Further, the flow-diagram to demonstrate the proposed methodology has been shown in Fig. 4.

The process of designing the proposed hybrid model comprises of two steps. The first step is creating a CAE network that reduces the dimensionality of the input leaf images from 256 × 256 to 32 × 32. The architecture of the CAE network is shown in Fig. 5 and its layer-wise details is tabulated in Table 2. The CAE network also contains a Decoder Network that is used to reconstruct the original data (in this case, leaf images) from the encoded data. The training of CAE network is done such that the Reconstruction Loss is minimized (discussed in Section 3.1.2). This ensures that the CAE network reduces the dimensionality of the leaf images without losing its important features. In this research work, *Normalized Root Mean Squared Error (NRMSE)* (Feng et al., 2015) loss function is used to compute the Reconstruction Loss between the original leaf image and the reconstructed leaf images. The formula to compute *NRMSE* is shown in Eq. (8).

$$NRMSE\left(X^O, X^R\right) = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(X^O - X^R\right)^2}}{P_{maximum} - P_{minimum}} \tag{8}$$

where X^O is the original leaf image, X^R is the leaf image reconstructed by the CAE network, N is the total number of leaf images taken into consideration, $P_{maximum}$ is the maximum pixel intensity in the input leaf images i.e., 255, and $P_{minimum}$ is the minimum pixel intensity in the input leaf images, i.e., 0.

Table 3
Layer wise details of the proposed hybrid model.

Layer number	Layer (type)	Input shape	Number of filters/channels	Size of each filter/channel	Activation function	Padding	Output shape	Number of training parameters
1	Input Layer	256 × 256 × 3	3	–	–	–	256 × 256 × 3	0
2	Conv #1	256 × 256 × 3	16	3 × 3	ReLu	Same Padding	256 × 256 × 16	448
3	MaxPool #1	256 × 256 × 16	16	2 × 2	–	–	128 × 128 × 16	0
4	Conv #2	128 × 128 × 16	8	3 × 3	ReLu	Same Padding	128 × 128 × 8	1,160
5	MaxPool #2	128 × 128 × 8	8	2 × 2	–	–	64 × 64 × 8	0
6	Conv #3	64 × 64 × 8	8	3 × 3	ReLu	Same Padding	64 × 64 × 8	584
7	MaxPool #3	64 × 64 × 8	8	2 × 2	–	–	32 × 32 × 8	0
8	Bottleneck Layer	32 × 32 × 8	8	3 × 3	ReLu	Same Padding	32 × 32 × 8	584
9	Conv #7	32 × 32 × 8	6	3 × 3	ReLu	Valid padding	30 × 30 × 6	438
10	MaxPool #4	30 × 30 × 6	6	2 × 2	–	–	15 × 15 × 6	0
11	Conv #8	15 × 15 × 6	16	3 × 3	ReLu	Valid padding	13 × 13 × 16	880
12	MaxPool # 5	13 × 13 × 16	16	2 × 2	–	–	6 × 6 × 16	0
13	Conv #9	6 × 6 × 16	16	3 × 3	ReLu	Valid padding	4 × 4 × 16	2,320
14	MaxPool # 6	4 × 4 × 16	16	2 × 2	–	–	2 × 2 × 16	0
15	Flatten Layer	2 × 2 × 16	–	–	–	–	64	0
16	Dense #1	64	–	–	ReLu	–	32	2,080
17	Dense #2	32	–	–	Sigmoid	–	1	33
Total number of trainable parameters								**5,751**
Total number of non-trainable parameters								**2776**

Fig. 8. (Top row) healthy leaf, (Bottom row) diseased leaf (Bacterial Spot).

After reducing the dimensionality of input leaf images, the CNN has been applied to classify the input leaf images as either a diseased leaf or a healthy leaf. The output of the Bottleneck Layer of the CAE is taken as the input for the CNN. Fig. 6 depicts the architecture of CNN that is used to classify leaf images.

As already discussed, the proposed hybrid model has been created by concatenating the layers of the encoder network of CAE and the layers of the CNN. The architecture of the hybrid model is shown in Fig. 7 and its layer-wise details are shown in Table 3.

The computation of training parameters for each Convolutional layer present either in the CAE network or in the proposed hybrid model is done by Eq. (9) (Goodfellow et al., 2016).

$$N^{i}_{training\ parameters} = \left(l^{i}_{filter} \times w^{i}_{filter} \times n^{i-1}_{channels} \times n^{i}_{filters}\right) + n^{i}_{filters} \tag{9}$$

where $N^{i}_{training\ parameters}$ is the number of training parameters in the i^{th} Convolutional layer present either in the CAE network or in the proposed model, l^{i}_{filter} and w^{i}_{filter} are the length and width of each Convolutional filters applied in the i^{th} Convolutional layer, respectively, $n^{i-1}_{channels}$ represent the number of channels present in the output of the $i-1^{th}$ layer, and $n^{i}_{filters}$ symbolize the number of convolutional filters applied in the i^{th} layer of the model.

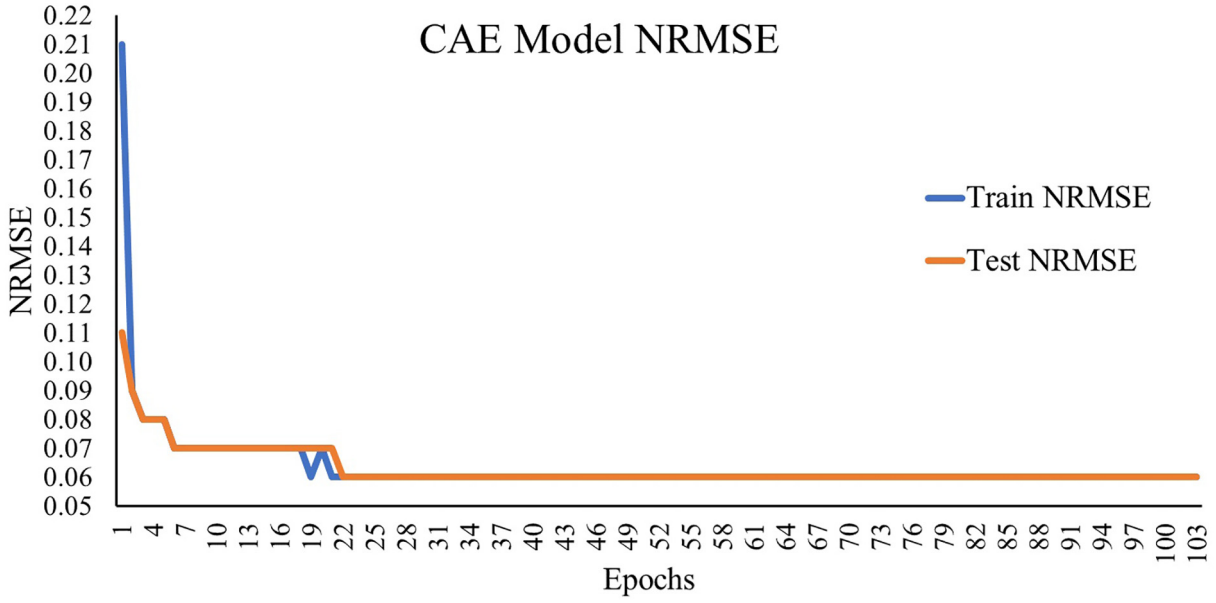


Fig. 9. Change in reconstruction loss with respect to epochs.

Fig. 10. (Top row) original leaf images, (Bottom row) reconstructed leaf images using the convolutional autoencoder network.

The fully connected layers have also been used at the end of the proposed model. The training parameter's calculation for the fully connected layers is done using Eq. (10) (Haykin, 1998).

$$N^i_{training\ parameters} = \left(n^i_{input} \times n^i_{output}\right) + n^i_{output} \tag{10}$$

where
$N^i_{training\ parameters}$ is the number of training parameters in the i^{th} Fully Connected layer, and n^i_{input} and n^i_{output} represent input and output size of i^{th} Fully Connected layer, respectively.

As already discussed, that the layers of the encoder network of CAE (i.e., layer 1 to layer 8) and layers of CNN (i.e., layer 8 to layer 17) has been used to create the proposed hybrid model. Since the layers imported from the encoder network of CAE are pre-trained therefore, training parameters of these layers are not included while computing the training parameters of the hybrid model. Hence it can be observed from Table 3 that the hybrid model uses only 5,751 trainable parameters and 2776 non-trainable parameters. Moreover, it can also be observed from Table 2 that the CAE network uses 4,163 training parameters. Thus, the total trainable parameters used in the proposed work can be calculated as the sum of trainable parameters of CAE and hybrid model. Since CAE uses 4,163 trainable parameters, and the hybrid model uses 5,751 trainable parameters so the total trainable parameters used in this research work is 9,914.

3.3. Dataset description

The proposed hybrid model has been applied to detect Bacterial Spot disease in peach plants. The leaf images of peach plants have been extracted from the PlantVillage dataset (Hughes and Salathe, 2015). The dataset contains 4457 leaf images of peach plants, which are evenly distributed in two classes: healthy and diseased (Bacterial Spot). The healthy class contains 2160 peach leaf images, and the diseased (Bacterial Spot) class comprise of 2297 leaf images of the peach plant. An example of a healthy and a diseased leaf image of a peach plant is shown in Fig. 8.

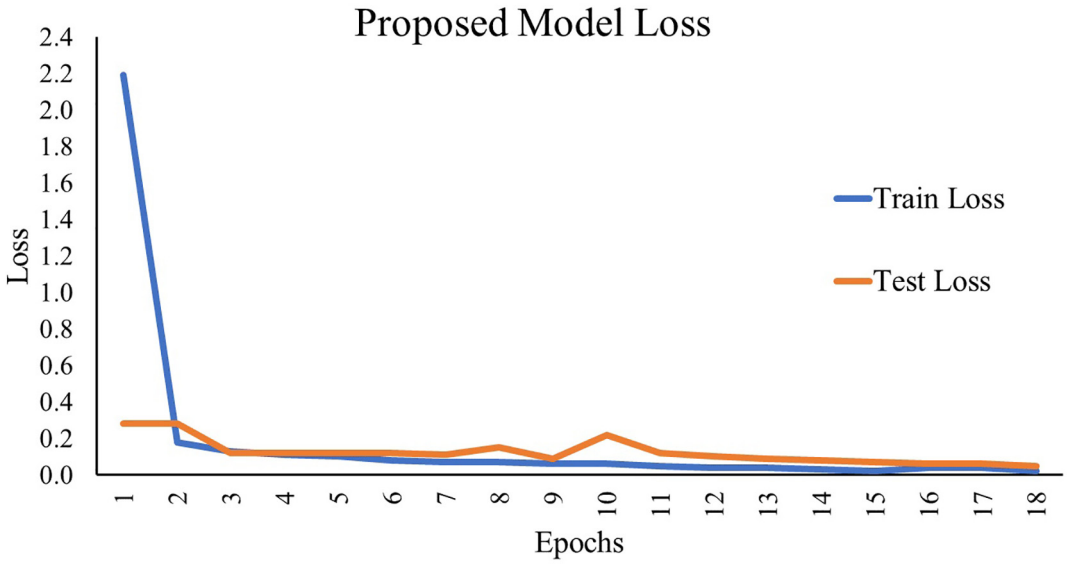


Fig. 11. Change in model loss with respect to epochs of the proposed hybrid model.

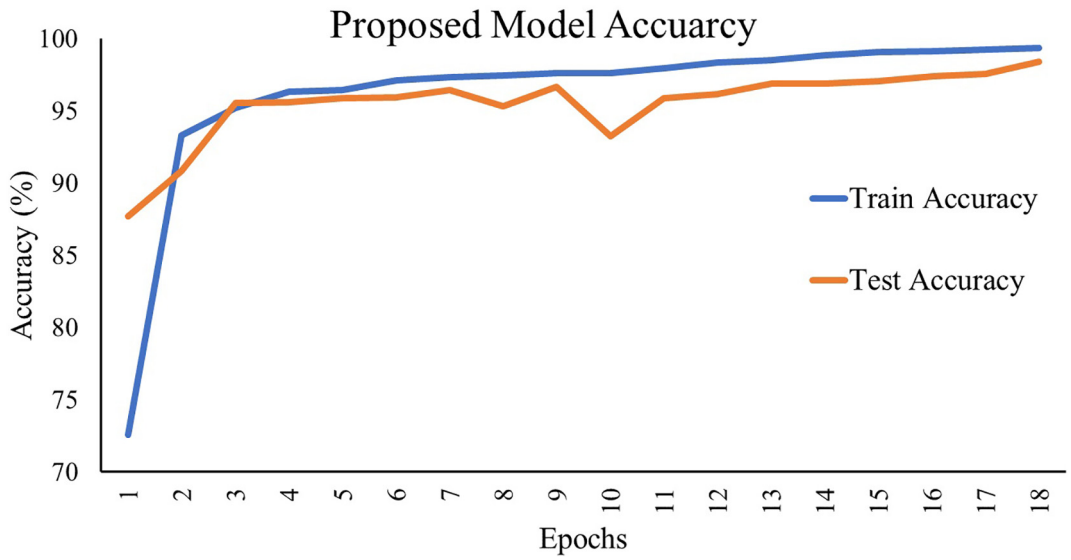


Fig. 12. Change in model accuracy with respect to epochs of the proposed hybrid model.

To train the model, the leaf images of peach plants have been randomly divided such that 70% of them form the training dataset, and 30% of them form the testing dataset. Thus, the training dataset has 3342 leaf images, and the testing dataset has 1115 leaf images.

3.4. Experimental configuration

The experiments performed in this research work uses the Jupyter Notebook, which is an Integrated Development Environment (IDE) for Python programming language. One can also use other programming languages such as Matlab, R, etc., to implement the proposed hybrid model. The *train_test_split* function of *sklearn* (Pedregosa et al., 2011) Application Programming Interface (API) of Python has been used to form the training and testing dataset. To create and train the model, the *Keras* API has been used.

To train the CAE network, Adam (Kingma and Ba, 2014) optimizer and NRMSE loss (Feng et al., 2015) have been used with batch size of 32 and 200 epochs. Similarly, Adam (Kingma and Ba, 2014) optimizer and Binary Cross-Entropy (BCE) loss have been used to train the proposed hybrid model with batch size of 32 and 100 epochs. In order to avoid retraining of layers imported from the CAE network, i.e., layer 1 to layer 8 (described in Table 2), False value has been assigned to the trainable flag of these layers during the training phase of the proposed hybrid model. To prevent model overfitting, Early stopping has been used with patience value equals 5 (i.e., if the testing loss does not improve over five consecutive epochs, then the training will stop).

4. Results and discussion

This section discusses the results of experiments performed in the current research work. First, the results of the CAE network are shown. Next, the results of the proposed hybrid model are shown.

The *NRMSE* loss has been used to evaluate the performance of the CAE network. It is computed with the help of original leaf images and reconstructed leaf images using Eq. (8). The train and test *NRMSE* loss between the original and the reconstructed leaf images are 0.0597 and 0.0607, respectively. The change in Reconstruction Loss with respect to epochs is shown in Fig. 9. Also, some original leaf images with their corresponding reconstructed leaf images are shown in Fig. 10.

The proposed hybrid model has achieved a training accuracy of 99.35% and a testing accuracy of 98.38%, with 0.02 training loss and 0.05 testing loss. The change of training and testing loss, along with training and testing accuracy with respect to epochs, are shown in Figs. 11 and 12, respectively.

Precision, Recall, and F1-measure (Zaki and Meira Jr, 2014) have also been computed for the proposed model. The proposed hybrid model has achieved 98.0% Precision. Its Recall is 98.72%, and its F1-measure is 98.36%. Table 4 presents the comparison of the proposed work with the various research works done in the literature.

From Table 4, it can be observed that the proposed hybrid model achieved 98.38% testing accuracy, which is more than the testing accuracies in the research works done by (Khamparia et al., 2020) with testing accuracy 86.78%, (Tiwari et al., 2020) with testing accuracy 97.8%, (Chohan et al., 2020) with testing accuracy 98%, and (Mohameth et al., 2020) with testing accuracy 98%. The testing accuracy of the proposed model is slightly lesser than the testing accuracies in the research

Table 4
Comparison of different state-of-the-arts with the proposed work based on their testing accuracies and number of training parameters.

Author(s) name and year	Proposed approach	Testing accuracy	Number of training parameters (approximately)
Khamparia et al. (2020)	Convolutional Encoder Network	86.78%	3.3 million
Sanga et al. (2020)	ResNet-152	99.2%	60 million
Tiwari et al. (2020)	VGG-19 + SVM	97.8%	143 million
Mohameth et al. (2020)	ResNet-50 + SVM	98%	25 million
Chohan et al. (2020)	VGG-19	98%	143 million
Ferentinos (2018)	VGGNet	99.5%	138 million
Mohanty et al. (2016)	GoogLeNet	99.3%	7 million
Proposed approach	**CAE + CNN**	**98.38%**	**9,914**

works done by (Sanga et al., 2020) with the testing accuracy 99.2%, (Mohanty et al., 2016) with the testing accuracy 99.3%, and (Ferentinos, 2018) with testing accuracy 99.5%. But, in the proposed work only 9,914 training parameters are used, which is very less as compared to the number of training parameters in the state-of-the-art systems discussed in Section 2 and tabulated in Table 4. Therefore, the proposed hybrid model requires very less training time and very less prediction time.

The proposed model has two prominent use cases. First, it can be trained and used for automatic plant disease detection on low-computational power systems with less training time and prediction time. Second, the proposed model can also be trained and used on smartphones. Running a Deep Learning model in mobile applications instead of sending the leaf images of plants to the cloud/server reduces the latency and provides data privacy to farmers.

5. Conclusion

Disease detection in plants at the early stages is a hard and challenging task. Many researchers have used different Machine Learning and Deep Learning techniques for automatic plant disease detection. However, most of these techniques either use millions of training parameters or have a low classification accuracy. In this paper, a novel hybrid model was proposed for automatic plant disease detection that was based on two Deep Learning techniques named Convolutional Autoencoder (CAE) network and Convolutional Neural Network (CNN). The proposed hybrid model first obtained compressed domain representations of leaf images using the encoder network of CAE and then used the compressed domain representations for classification using CNN. Due to dimensionality reduction using CAE, the number of features, and hence the number of training parameters reduced significantly as compared to existing state-of-the-art systems. To test the model, it was applied to detect Bacterial Spot disease in peach plants. The model achieved 99.35% training accuracy and 98.38% testing accuracy by using only 9,914 training parameters. Fewer training parameters used in the proposed hybrid model significantly decreased the time required to train the model for automatic plant disease detection and the time required to identify the disease in plants using the trained model.

Funding source

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Ahmed, K., Shahidi, T.R., Irfanul Alam, S.M., Momen, S., 2019. Rice leaf disease detection using machine learning techniques. 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI). IEEE, Dhaka, Bangladesh, pp. 1–5.

Ashraf, T., Khan, Y.N., 2020. Weed density classification in rice crop using computer vision. Comput. Electron. Agric. 175, 105590. https://doi.org/10.1016/j.compag.2020.105590.

Bisong, E., 2019. Autoencoders. Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners. Apress, Berkeley, CA, pp. 475–482. https://doi.org/10.1007/978-1-4842-4470-8_37.

Chen, Junde, Chen, Jinxiu, Zhang, D., Sun, Y., Nanehkaran, Y.A., 2020. Using deep transfer learning for image-based plant disease identification. Comput. Electron. Agric. 173, 105393. https://doi.org/10.1016/j.compag.2020.105393.

Chohan, M., Khan, A., Katper, S., Mahar, M., 2020. Plant disease detection using deep learning. Int. J. Recent Technol. Eng. 9 (1), 909–914. https://doi.org/10.35940/ijrte.A2139.059120.

Es-saady, Y., el Massi, I., el Yassa, M., Mammass, D., Benazoun, A., 2016. Automatic recognition of plant leaves diseases based on serial combination of two SVM classifiers. 2016 International Conference on Electrical and Information Technologies (ICEIT). IEEE, Tangiers, Morocco, pp. 561–566. https://doi.org/10.1109/EITech.2016.7519661.

Feng, D., Feng, M., Ozer, E., Fukuda, Y., 2015. A vision-based sensor for noncontact structural displacement measurement. Sensors 15 (7), 16557–16575. https://doi.org/10.3390/s150716557.

Ferentinos, K.P., 2018. Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agric. 145, 311–318. https://doi.org/10.1016/j.compag.2018.01.009.

Golhani, K., Balasundram, S.K., Vadamalai, G., Pradhan, B., 2018. A review of neural networks in plant disease detection using hyperspectral data. Inform. Proc. Agric. 5 (3), 354–371. https://doi.org/10.1016/j.inpa.2018.05.002.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. 1st ed. The MIT Press, Cambridge, United Kingdom.

Haykin, S., 1998. Neural Networks: A Comprehensive Foundation. 2nd ed. Prentice Hall PTR, USA.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Las Vegas, NV, USA, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.

Himani, 2014. An analysis of agriculture sector in Indian economy. IOSR J. Human. Soc. Sci. 19 (1), 47–54. https://doi.org/10.9790/0837-191104754.

Hughes, David P., Salathe, M., 2015. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv 1–13 preprint arXiv.

Islam, M., Dinh, Anh, Wahid, K., Bhowmik, P., 2017. Detection of potato diseases using image segmentation and multiclass support vector machine. 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE). IEEE, Windsor, ON, Canada, pp. 1–4. https://doi.org/10.1109/CCECE.2017.7946594.

Jearanaiwongkul, W., Anutariya, C., Andres, F., 2018. An ontology-based approach to plant disease identification system. Proceedings of the 10th International Conference on Advances in Information Technology - IAIT 2018. ACM Press, New York, New York, USA, pp. 1–8. https://doi.org/10.1145/3291280.3291786.

Karlekar, A., Seal, A., 2020. SoyNet: soybean leaf diseases classification. Comput. Electron. Agric. 172, 105342. https://doi.org/10.1016/j.compag.2020.105342.

Khamparia, A., Saini, G., Gupta, D., Khanna, A., Tiwari, S., de Albuquerque, V.H.C., 2020. Seasonal crops disease prediction and classification using deep convolutional encoder network. Circ. Syst. Sign. Proc. 39, 818–836. https://doi.org/10.1007/s00034-019-01041-0.

Kim, W.-S., Lee, D.-H., Kim, Y.-J., 2020. Machine vision-based automatic disease symptom detection of onion downy mildew. Comput. Electron. Agric. 168, 105099. https://doi.org/10.1016/j.compag.2019.105099.

Kingma, D., Ba, J., 2014. Adam: a method for stochastic optimization. International Conference on Learning Representations. San Diego, CA, USA, pp. 1–15.

Krithika, N., Selvarani, A.G., 2017. An individual grape leaf disease identification using leaf skeletons and KNN classification. 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS). IEEE, Coimbatore, India, pp. 1–5 https://doi.org/10.1109/ICIIECS.2017.8275951.

Marwaha, S., Bedi, P., Yadav, R., Malik, N., 2009. Diseases and pests identification in crops - a semantic web approach. Proceedings of the 4th Indian International Conference on Artificial Intelligence, IICAI 2009. IICAI, Tumkur, Karnataka, India, pp. 1057–1076.

Marwaha, S., Chand, S., Saha, A., 2012. Disease diagnosis in crops using content based image retrieval. 12th International Conference on Intelligent Systems Design and Applications (ISDA). IEEE, Kochi, India, pp. 729–733. https://doi.org/10.1109/ISDA.2012.6416627.

Mohameth, F., Bingcai, C., Sada, K.A., 2020. Plant disease detection with deep learning and feature extraction using Plant Village. J. Comp. Commun. 8 (6), 10–22. https://doi.org/10.4236/jcc.2020.86002.

Mohanty, S.P., Hughes, D.P., Salathé, M., 2016. Using deep learning for image-based plant disease detection. Front. Plant Sci. 7, 1–10. https://doi.org/10.3389/fpls.2016.01419.

Naik, M.R., Sivappagari, C.M.R., 2016. Plant leaf and disease detection by using HSV features and SVM classifier. Int. J. Eng. Sci. 6 (12), 1–4.

Ngugi, L.C., Abelwahab, M., Abo-Zahhad, M., 2020. Recent advances in image processing techniques for automated leaf pest and disease recognition – a review. Inform. Proc. Agric., 1–25. https://doi.org/10.1016/j.inpa.2020.04.004.

Padol, P.B., Yadav, A.A., 2016. SVM classifier based grape leaf disease detection, in: 2016 Conference on Advances in Signal Processing (CASP). IEEE, Pune, India, pp. 175–179 https://doi.org/10.1109/CASP.2016.7746160.

Panda, Panigrahi Kshyanaprava, Himansu, Das, Kumar, Sahoo Abhaya, Chandra, Moharana Suresh, 2020. Maize leaf disease detection and classification using machine learning algorithms. Progress in Computing. Analytics and Networking. Springer Singapore, Singapore, pp. 659–669.

Pardede, H.F., Suryawati, E., Sustika, R., Zilvan, V., 2018. Unsupervised convolutional autoencoder-based feature learning for automatic detection of plant diseases. 2018 International Conference on Computer, Control, Informatics and its Applications (IC3INA). IEEE, Tangerang, Indonesia, Indonesia, pp. 158–162. https://doi.org/10.1109/IC3INA.2018.8629518.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Qin, F., Liu, D., Sun, B., Ruan, L., Ma, Z., Wang, H., 2016. Identification of alfalfa leaf diseases using image recognition technology. PLoS One 11 (12), 1–26. https://doi.org/10.1371/journal.pone.0168274.

Ramesh, S., Vydeki, D., 2020. Recognition and classification of paddy leaf diseases using optimized deep neural network with Jaya algorithm. Inform. Proc. Agric. 7 (2), 249–260. https://doi.org/10.1016/j.inpa.2019.09.002.

Rao, D.R., Krishna, M., Ramakrishna, B., 2020. Smart ailment identification system for Paddy crop using machine learning. Int. J. Innov. Eng. Manag. Res. 9 (3), 96–100.

Sanga, S.L., Machuve, D., Jomanga, K., 2020. Mobile-based deep learning models for Banana disease detection. Technol. Appl. Sci. Res. 10 (3), 5674–5677.

Sharma, P., Berwal, Y.P.S., Ghai, W., 2019. Performance analysis of deep learning CNN models for disease detection in plants using image segmentation. Inform. Proc. Agric., 1–9. https://doi.org/10.1016/j.inpa.2019.11.001.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. arXiv 1–14 preprint arXiv.

Singh, V., 2019. Sunflower leaf diseases detection using image segmentation based on particle swarm optimization. Artific. Intellig. Agric. 3, 62–68. https://doi.org/10.1016/j.aiia.2019.09.002.

Singh, V., Misra, A.K., 2017. Detection of plant leaf diseases using image segmentation and soft computing techniques. Inform. Proc. Agric. 4, 41–49. https://doi.org/10.1016/j.inpa.2016.10.005.

Szegedy, C., Liu, Wei, Jia, Yangqing, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Boston, MA, USA, pp. 1–9. https://doi.org/10.1109/CVPR.2015.7298594.

Tewari, V.K., Pareek, C.M., Lal, G., Dhruw, L.K., Singh, N., 2020. Image processing based real-time variable-rate chemical spraying system for disease control in paddy crop. Artific. Intellig. Agric. 4, 21–30. https://doi.org/10.1016/j.aiia.2020.01.002.

Tiwari, D., Ashish, M., Gangwar, N., Sharma, A., Patel, S., Bhardwaj, S., 2020. Potato leaf diseases detection using deep learning. 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, Madurai, India, India, pp. 461–466. https://doi.org/10.1109/ICICCS48265.2020.9121067.

Zaki, M.J., Meira Jr., Wagner, 2014. Data Mining and Analysis: Fundamental Concepts and Algorithms. 2nd ed. Cambridge University Press, Cambridge, United Kingdom.