Pertanika J. Sci. & Technol. 31 (2): 813 - 841 (2023)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Transfer Learning VGG16 Model for Classification of Tomato Plant Leaf Diseases: A Novel Approach for Multi-Level Dimensional Reduction

Premkumar Borugadda*, Ramasami Lakshmi and Satyasangram Sahoo

Department of Computer Science, School of Engineering & Technology, Pondicherry University, Karaikal Campus, Puducherry-609605, India

ABSTRACT

Tomato is the most popular and cultivated crop in the world. Nevertheless, the quality and quantity of tomato crops have been declining due to various diseases that afflict tomato crops. Hence, it becomes necessary to detect the disease early to prevent crop damage and increase the yield. The proposed model in this article predicts the infected tomato leaf images (9 classified diseases and also healthy class) obtained from the Plant Village dataset. In this model, Transfer learning was used to extract features from images by VGG16, yielding a high dimension of 25088 features. Overfitting is a commonly anticipated problem because of the higher dimensionality of data. To mitigate this problem, the authors have adopted a novel dimensional reduction-based technique: filter methods, feature extraction techniques like Principal Components Analysis (PCA), and the Boruta feature selection technique of wrapper methods. This adoption enables the proposed model to attain a significantly improved high accuracy of 95.68% and 95.79% in MLP and VGG16, respectively, by reducing its initial dimension on the tomato dataset containing 18160 images across 10 classes.

Keywords: Boruta algorithm, filter methods, plant leaves dataset, principal component analysis, tomato leaf disease classification, VGG16

ARTICLE INFO

Article history:
Received: 21 May 2022
Accepted: 16 August 2022
Published: 06 March 2023

DOI: https://doi.org/10.47836/pjst.31.2.09

E-mail addresses:
premkumar.jones@gmail.com (Premkumar Borugadda)
prof.rlakshmi@gmail.com (Ramasami Lakshmi)
smartlincoln@gmail.com (Satyasangram Sahoo)
*Corresponding author

INTRODUCTION

Tomato (*Lycopersicon esculentum*) is an extensively farmed agricultural crop. This crop's growing season lasts about 90 to 150 days, with typical daytime average temperatures of 18 to 25°C and nighttime temperatures of 10 to 20°C (Gadekallu et al., 2021). Excess humidity

ISSN: 0128-7680
e-ISSN: 2231-8526

© Universiti Putra Malaysia Press

and reduced sunlight exposure negatively impact crop quality. Generally, excess humidity renders crops vulnerable to pests, diseases, and decay. Hence a dry climate is necessary for producing high-quality tomatoes. Crops ought to be closely monitored for signs of fungi, bacteria, and virus invasion. Necessary preventive actions should be in place to check and control diseases early. It can be achieved by roping in experts and field workers to detect and identify diseases at an early stage, albeit the task might prove to be laborious and economically unviable when there is high acreage involved. It becomes essential to make use of technology that could significantly reduce human intervention for performing real-time monitoring with the highest precision possible. The recent developments in the deep learning (DL) models make it possible to detect and identify diseases in tomato plants at an early stage.

Precision farming may be used to combat diseases and pests that affect crops. Sensor networks, remote sensing, robotics, computer vision, machine learning, and DL are employed in precision farming. Computer vision forms an integral part of precision agriculture. For plant disease recognition and classification in agriculture, computer vision DL-based algorithms have been applied. Various studies have used DL agriculture models to diagnose crop problems (Guo et al., 2003). DL networks contain several layers of complex formation that increase the model's accuracy. Nodes of one layer are interconnected with another node of layers to form classification-based architectures and require additional computational power for training. Convolution neural networks (CNNs) are broadly used in the DL architecture framework. CNN is involved in many applications, such as image classification and object recognition (Tang & Wu, 2016) which significantly improves image classification in several fields, such as agriculture. The present study proposed a multi-level dimension reduction (filter methods, principal component analysis (PCA) and Boruta feature selection) method to obtain optimal features and classify tomato plant leaf diseases using VGG16, multi-layer perceptron (MLP), and machine learning algorithms (MLA). This study might help farmers identify the diseases early and prevent loss so that the crop yield would increase. The tomato leaf dataset for the present study was obtained from the plant village dataset (Mohanty et al., 2016). The framework was used to classify the infected and healthy images of the tomato leaves. The performance of CNN-based models, VGG16, MLP, and MLA, was analyzed based on different evaluation metrics, such as training accuracy, validation accuracy, and weighted average F1 score.

This study primarily focused on the declined dimension in a multi-level model, as extensive data on the number of samples and features from images were collected. Classifying the images with the extracted features from a high-dimensional feature vector is critical. In some cases, the number of features (F) is more compared to the number of samples (S) (F>S), which is known as the curse of dimensionality (CoD). Due to the huge dimension, the dimensionality of the image data needs to be reduced using

dimension reduction techniques that help with image classification without losing the most significant information. Hence, this study focused primarily on addressing the high dimensionality of data. The machine learning (ML) or DL models were trained effectively on high dimensionality data; however, the models encounter problems, such as overfitting, requiring more training time, consumption of significant resources, high model complexity, containing trainable parameters, and taking up a large amount of storage space. A multi-level dimension reduction algorithm was proposed in this study to overcome these issues. It consisted of multi-level dimension reduction methods, such as filter methods of feature selection in dimension reduction as level 1, principal components analysis (PCA) of feature transformation as level 2, and Boruta wrapper method as level 3.

REVIEW OF LITERATURE

Computer vision with data science is a trending technology in agriculture for the classification of the disease of plants. In the early days, traditional MLA was used to identify plant leaf diseases in agriculture as machine learning methods were unable to classify the images due to large-scale image datasets.

Durmuş et al. (2017) trained AlexNet (Krizhevsky et al., 2012) and SqueezeNet (Iandola et al., 2016) on the 18160 tomato leaf images taken from the plant village dataset and classified the tomato diseases using the supercomputer Nvidia Jetson Tx1; the training and validation presented an accuracy of 94.3% and 95.65% using SqueezeNet and AlexNet, respectively.

Tm et al. (2018) proposed an approach that includes data acquisition, preprocessing and classification. In the present study, a variation of LeNet was applied to the tomato dataset. It consisted of approximately 18160 images belonging to ten different classes of tomato leaf diseases. Keras, a neural network API (Application Programming Interface) written in Python, has been used for the model implementation. The highest validation accuracy of 94.8% was obtained over 30 epochs of training.

Durmus et al. (2017) proposed deep-CNNs, such as ResNet50, for tomato leaf disease detection using PyTorch. A DL technique with transformation and augmentation was used to overcome the overfitting problem and improve the model's performance. The proposed model yielded 97% accuracy after fine-tuning the weights for the ResNet model.

Gadekallu et al. (2021) proposed a novel PCA-whale optimization algorithm (WOA) hybrid optimization technique for significant features extracted from the 18160 tomato leaf images. The deep neural network was trained on the optimal features with 94% accuracy. The present study used multi-level three dimension reduction techniques (filter method, feature transformation (PCA), and Boruta of wrapper methods) to obtain the optimal features. The final results improved and were compared to the previous results on the same number of tomato leaf images in the tomato dataset.

The objective of this study was to reduce the high-dimension features into optimal features using a multi-level dimension reduction algorithm. With the high dimension of data, problems, such as overfitting, high training time, model complexity, trainable parameters, and large storage space of the model, occurred. Thus, to prevent all these problems and improve accuracy, the study proposes three types of dimension reduction techniques to build an efficient agricultural tomato leaf disease classification model.

METHODOLOGY

The detailed methodology for the tomato leaf disease classification framework is given below as follows:

Figure 1 illustrates the classification framework consisting of six stages: data acquisition, preprocessing data, feature extraction stage, dimensionality reduction stage, classification stage, and selecting the optimal prediction model.

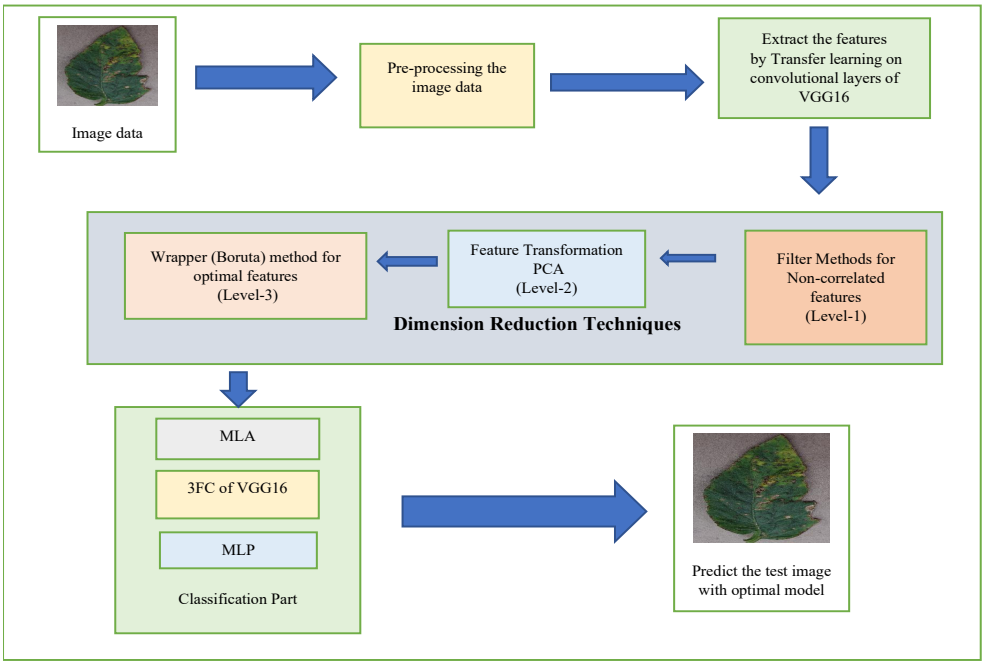


Figure 1. Architecture framework for classification of tomato leaf images

Dataset

In the initial stage, the images of tomato disease were taken from the plant village dataset (Mohanty et al., 2016). A total of 54,345 photos of 14 crops were included in this plant village collection. These crops include various fruits and vegetables, such as apples and blueberries. Images of tomato leaves were utilized in this investigation. Table 1 shows the

number of images for each class of tomato leaves, as illustrated in Figure 2. Ten categories of tomato images, including those considered healthy, were available. In the present study, 18160 images of the tomato dataset were split in the ratio of 90:05:05 for training, validation, and testing, respectively.

Table 1
Dataset

S. No	Name of the class	No. of images
	Tomato__Bacterial spot	2127
	Tomato_Early_blight	1000
	Tomato__Late blight	1909
	Tomato_ Leaf Mold	952
	Tomato_Septoria_leaf_spot	1771
	Tomato_Spider_mites Two-spotted_spider_mite	1676
	Tomato___Target Spot	1404
	Tomato___Tomato_Yellow_Leaf_Curl_Virus	5357
	Tomato___Tomato_mosaic_virus	373
	Tomato___healthy	1591
	Total number of images	18160

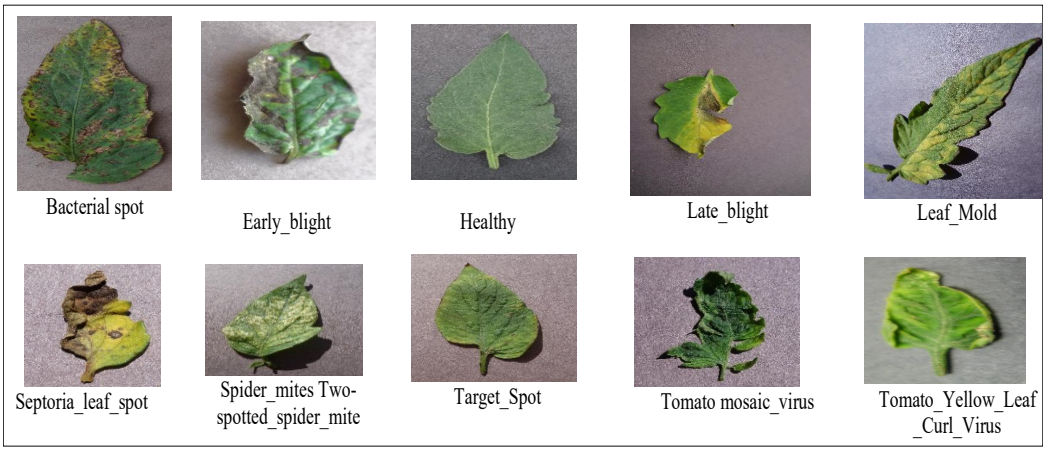


Figure 2. Sample tomato leaf images of different classes of the plant village dataset

Preprocessing

In the preprocessing, the size of the input image was 256×256 pixels, which was resized into 224×224 pixels. The labels of the image dataset were categorical. Thus, label encoding

(Cerda & Varoquaux, 2020) and one-hot encoding (Li et al., 2018) were applied for numerical values. The normalized pixel values of the image were placed between 0 and 1.

Feature Extraction

In the feature extraction stage, the standard VGG16 (Simonyan & Zisserman, 2014) model was applied to extract the features of image data, as shown in Figure 3. The two feature extraction methods were as follows: First, 13 convolutional layers of the VGG16 model were applied, and the high-dimensional features of the images were extracted, as shown in Figure 4. Second, the transfer learning method (Tammina, 2019) was applied to 13 convolutional layers of VGG16 for feature extraction (Figure 5).

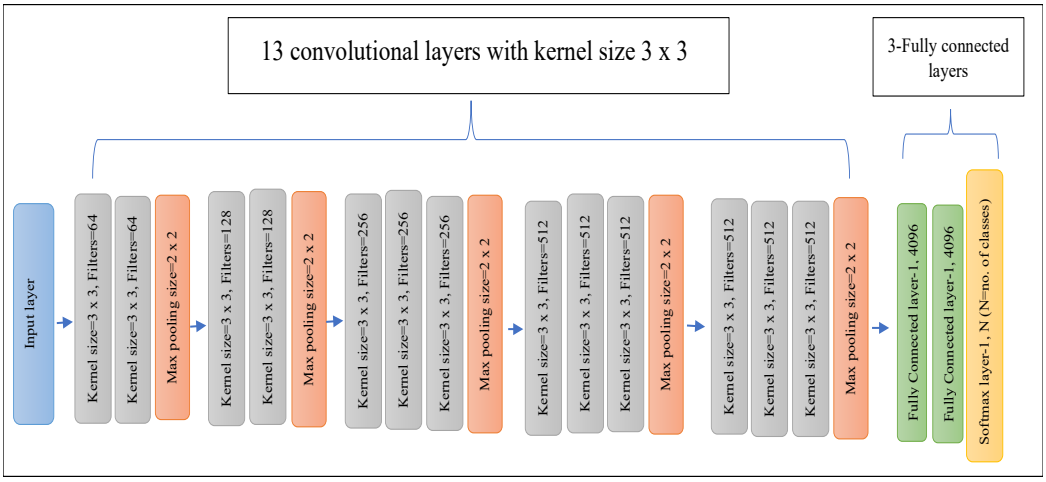


Figure 3. Standard VGG16 model architecture

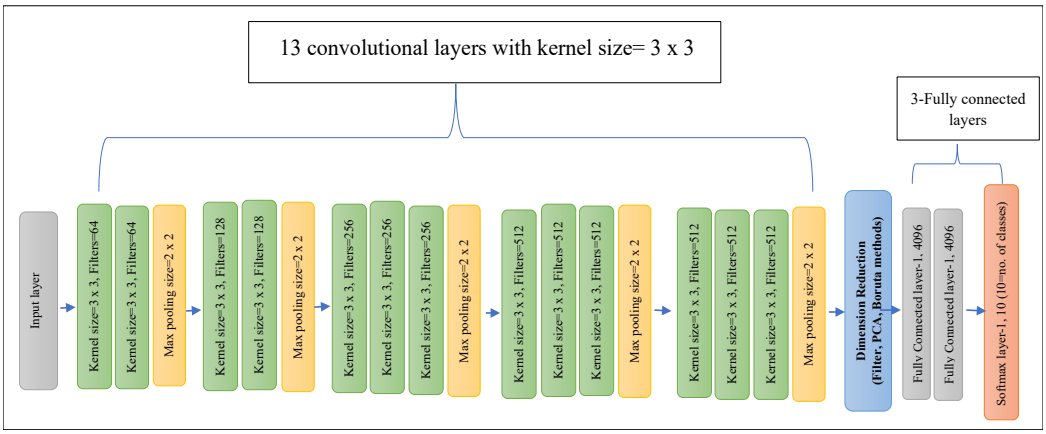


Figure 4. Applying dimension reduction techniques between 13 convolutional layers and 3 fully connected layers of VGG16

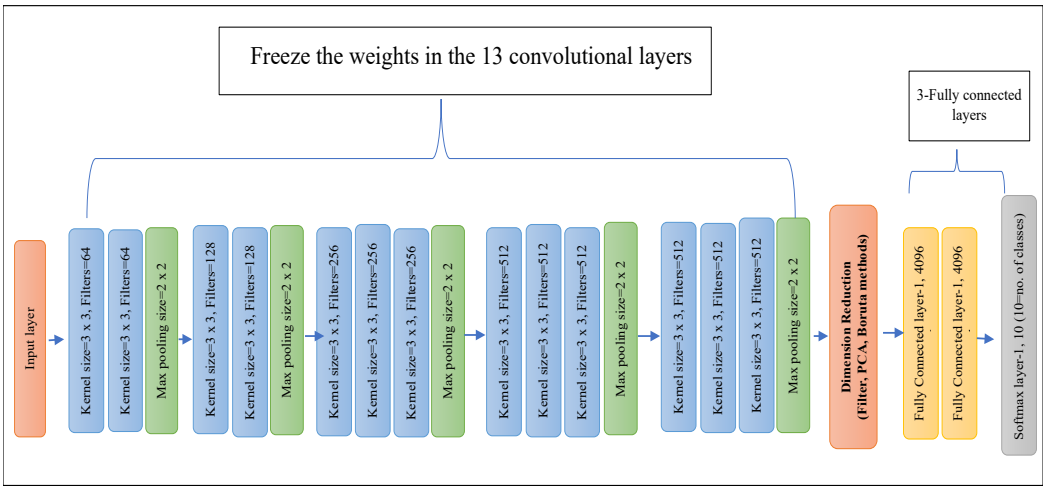


Figure 5. Applying dimension reduction methods between transfer learning on 13 convolutional layers of VGG16 and 3 fully connected layers

Dimension Reduction Technique

In the feature extraction stage, high dimensional features of 25088 were obtained with 13-convolutional layers of the pre-transfer learning stage on VGG16. If the classification models were trained using MLA, MLP, and three fully connected layers of VGG16 with high dimensional features, the models faced the following issues: the possibility that the model is biased towards overfitting, model computation will be high, and the performance of the models may be reduced due to curse of dimensionality. In order to overcome these problems, the model proposed a multi-level dimension reduction technique, such as filter, PCA, and Boruta, so that less significant features were eliminated from the images.

Filter Methods for Non-Correlated Features

Variance and correlation statistical methods were selected to obtain the optimal features.

Removing Constant Features

Constant features contain only one value, and the variance threshold value is 0. Totally of 994 constant features were identified from 25088 high-dimensional features (Doquire & Verleysen, 2013). However, the constant features do not affect the models and are removed, leaving 24094 high-dimensional features.

Removing Quasi-Constant Features

Quasi-constant features are similar to constant features with a variance threshold value of

0.01 (Ma et al., 2018). A total of 13175 quasi-constant features were identified and excluded from 24094 high dimensional features, retaining 10919 features.

Removing Correlated Features

Correlated features create redundancy and should be removed (Chuanlei et al., 2017). Three different threshold values, such as 0.6, 0.7, or 0.8, were typically applied to identify the correlated features. Depending on the dataset requirements, the threshold value of 0.8 was applied, and 136 correlated features were identified and eliminated. Finally, 10783 non-correlated features were identified in filter methods of dimension reduction level 1, as shown in Figure 6.

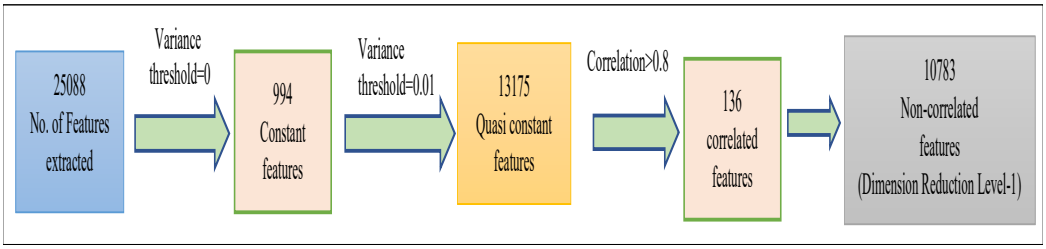


Figure 6. Filter method for non-correlated features at dimension reduction level 1

Application of the Feature Transfer Method-PCA

PCA was used for feature extraction. This method created new features by projecting existing features (Mudrova & Procházka, 2005). PCA is a dimensionality reduction technique to transform into a new lower-dimension dataset without losing critical information.

The dataset consisted of 'X' independent variables (dimensions) and one target variable. So, the total dimension size was 'X+1'.

Step 1: Standardize or scale the input dataset 'X' using Z-score.

$$Z = \frac{Xi - \bar{X}}{Std\ (X)} \tag{1}$$

Initially, the mean and standard deviation were calculated for each independent variable 'X'.

$$\bar{X} = \text{mean of X} = X = \frac{\sum_{i=1}^{n} X_i}{n} \tag{2}$$

Standardized value of X_i= (X_i- mean of X)/Standard deviation of X

$$\text{Std (x)=standard deviation of X} = \sqrt{\frac{\sum_{i=1}^{n}(X_i - x)^2}{n-1}} \tag{3}$$

Step 2: Covariance matrix of the scaled data without the target variable was calculated

$$\text{Cov(x,y)} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(y_i - \bar{y})}{n-1} \tag{4}$$

Step 3: Eigenvalues were computed

Step 4: Eigenvectors were defined
Step 5: Existing input datasets were projected into new dimensions using eigenvectors.

With 99% of the variance, PCA formed a new set of 5720 components from 10783 non-correlated features (Figure 7).

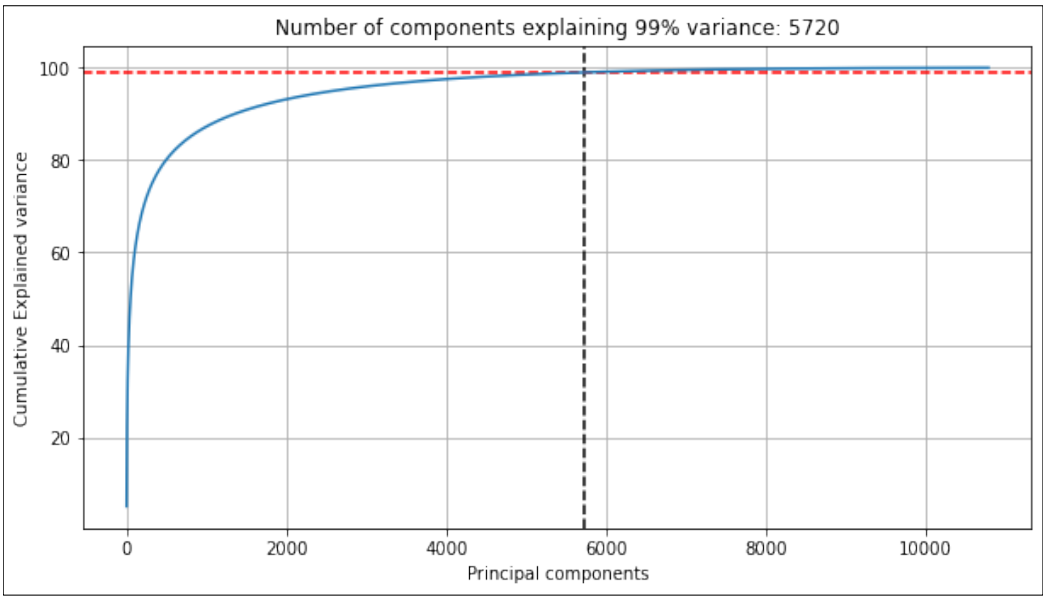


Figure 7. Number of components explained 99% of variance *vs.* cumulative explained variance

In Figure 7, principal components were represented on the horizontal axis and cumulative experience variance on the vertical axis at 99% of explained variance ratio; subsequently, 5720 principal components were obtained. In Figure 8, 10783 non-correlated features of dimension reduction level 1 were transformed into 5720 principal components.

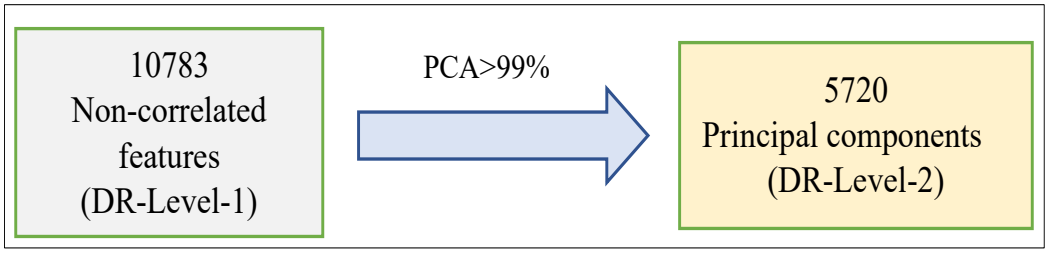


Figure 8. Conversion of non-correlated features at level 1 into principal components at level 2

Application of the Wrapper Methods (Boruta Feature Selection Algorithm)

The wrapper feature selection method identified optimal features using MLA (Chen & Chen, 2015). Herein, the random forest classification algorithm was used as a base model for the

Boruta algorithm (Kursa & Rudnicki, 2010). The algorithm consisted of the following steps:

1. In the first step, Boruta begins by cloning the supplied collection of original features. These were referred to as shadow features to distinguish them from the originals. The values of the shadow features were then rearranged to exclude the correlations.
2. In the second step, the significance features were validated using the mean decrease impurity (MDI), and the shadow features were trained using the random forest (RF) classifier. MDI determined each cloned feature's relevance.
3. The Z score was used to determine whether the original feature provided had a higher Z score than the maximum MDI score of the shadow feature.
4. 'Hits' were assigned to a vector with a high Z value. When the number of iterations was reached, a hit table was generated at the end of the process.
5. A feature with the highest Z score was marked as essential in each algorithm iteration. The final collection of features was derived from the hit vector.

All the procedures from the beginning to this point were repeated until the qualities of all the offered features were identified.

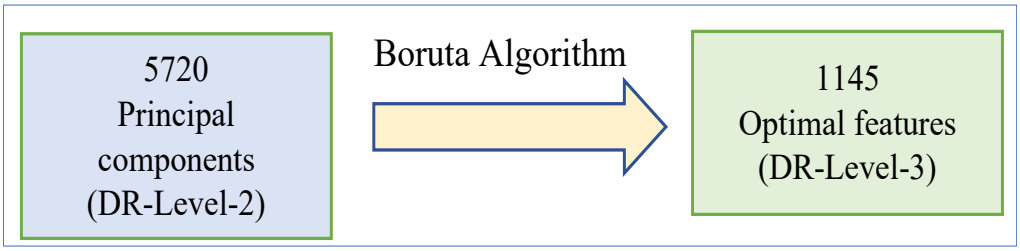


Figure 9. Conversion of principal components at level 2 into optimal features at level 3 with the Boruta algorithm

A total of 5720 principal components were obtained in dimension reduction level 2. Then, the Boruta feature selection algorithm was applied to these principal components, and 1145 optimal features were obtained at dimension reduction level 3 (Figure 9).

The proposed multi-level dimension reduction algorithm is shown in Figure 10. It represented an overview of the multi-level dimension reduction using a raw image as input; subsequently, optimal features were generated. The final feature was considered the reduced dimension available from the multi-level dimension, which was considered a maximum hit in the Boruta-based method.

Classification Algorithms

A total of 1145 optimal features were obtained in the Boruta feature selection algorithm. The classification algorithms were applied to these features.

```
Input: Images
 image ← resize.image
 dimension ← calculate_Dim(shape.Image)
 Initialize Reduced_Dim ← dimension
 Initialize Level ← None
If 'Reduced_Dim = = dimension' then:
    image.Weight  ←  image. VGG16(ImageNet)
    Reduced_Dim ← image.Weight
    add(level,1)
    Return(Reduced_Dim)
 If 'Reduced_Dim < dimension' & 'level= =1' then:
    Remove constant features:
          Set feature_filter. threshold = 0
          Constant_features ←  feature_filter (Reduced_Dim.shape)
          subtract (Reduced_Dim, Constant_features)
    Remove  Quasi_constant features:
          Set feature_filter. threshold = 0.01
          Quasi_features ← featurefilter(Reduced_Dim.shape)
          subtract(Reduced_Dim, Quasi_features)
     Remove correlated features:
          Set feature_filter.threshold = 0.8
          correlated_features ← features_filter(Reduced_Dim.shape)
          subtract(Reduced_Dim, correlated_features)
     add (level,1)
     Return (Reduced_Dim)
 If  'Reduced_dim < dimension' & 'level= =2' then:
      Set variance =99
      Components ← PCA(Reduced_dim, variance)
      Reduced_dim ← components
      add (level, 1)
      Return(Reduced_Dim)
 If 'Reduced_Dim < dimension' & 'level==3' then:
     Initialize n ← 0
     Max_Iteration ← max (n, 50)
     Repeat upto 'Max_Iteration' or 'Reduced_dim'
          cloned_dim[i] ← Reduced_dim[i]
          suffle.feature(cloned_dim[i])
          merge (Cloned_dim[i], Reduced_dim[i])
          Z_Score [i] = Radom_forest(reduced_dim)
          MDI_Score [i] = Random_forest(cloned_dim)
          Hit_count ← calculate (Z_Score [i] > max (MDI_Score[i]))
          Add (n, 100)
      Final_Features ← max (Hit_Count)
```

Figure 10. The Proposed multi-level dimension reduction algorithm

MLA

Classification is the process of organizing a given set of data into classes. Classification algorithms map the input data to classes, and the model predicts the input class. All MLA, such as logistic regression (Dreiseitl & Ohno-Machado, 2002), decision tree classifier (DTC) (Ali et al., 2012), random forest classifier (RFC) (Ali et al., 2012), Ada boost classifier (ABC) (Khammari et al., 2005), K Nearest Neighbor (KNN) (Zhang & Zhou, 2005), support vector classifier (SVC) (Awad & Khanna, 2015) and XG Boost (XGB) (Torlay et al., 2017) are not suitable for high dimension without any reduction techniques. Thus, we proposed a model of the eased task as a classifier. Finally, 1145 optimal features were fed to classification algorithms, the models were trained, and the results were compared with the above algorithms.

MLP

MLP is an example of an artificial neural network. It is applied broadly to solve several problems, such as pattern recognition and interpolation (Noriega, 2005). MLP is a neural network with fully connected multiple dense layers. The MLP network comprises input, hidden, and output layers that perform the computational work.

The neurons of the MLP are prepared to perform backpropagation learning for any classification and forecast job in the network. Several studies have applied different optimization methods for MLP. Ramchoun et al. (2016) presented a different approach for optimizing MLP architecture to instruct the network with a backpropagation algorithm. The inputs to the neuron (x) feature and the amounts in terms of weight (w) were computed for classification. The activation function (f) was included in the sum of the outcome to construct the output in an MLP. MLP network wherein the input layer contains nodes equal to the number of features extracted. The MLP's output layer uses softmax activation for multi-class classification, while the hidden layer uses relu activation functions. If a dataset has 'm' classes, the output layer uses 'm' nodes for model prediction.

Three Fully Connected Layers of VGG16

The VGG16 architecture consisted of 16 layers, of which 13 convolutional layers were explained in the feature extraction phase. The remaining three fully connected layers were used for classification purposes. 2/3 of the fully connected layers consisted of 4096 nodes, and the third fully connected layer had the softmax activation function for classification, which consisted of several labels (classes) (Gao & Pavel, 2017). In the present study dataset, ten classes were available.

RESULTS AND DISCUSSION

The performance measures classification models are described with respect to the experimental setup and hyperparameter tuning. Subsequently, the classification performance of the proposed model was evaluated and compared to the three fully connected layers of VGG16, MLP, and state-of-the-art MLA.

Performance Evolution

The performance of the classification models was evaluated using a confusion matrix, as shown in Figure 11. The evaluation parameters in the confusion matrix are accuracy, precision, recall (sensitivity), F1_Score, true-positive (TP_i), false-positive (FP_i), true-negative (TN_i), false-negative (FN_i), and class C_K. The subscript 'k' indicates the number of classes, and 'i' values from 0 to 'K'. PVC_1 means predicted values of class C_1, and AVC_1 means the actual values of class C_1. This study calculated precision, recall, and F1_Score over the validation dataset. The training and validation datasets were imbalanced. Thus, performance measures, such as accuracy, weighted-average-based precision, recall, and F1 score (Sokolov & Lapalme, 2009; Behera et al., 2019), were used to evaluate the performance of the classification models. The performance measurements of accuracy, W.A.P, W.A.R, and W.A.F1, were defined as follows.

$$\text{Accuracy} = \frac{\sum_{i=1}^{K} \frac{TP_i+TN_i}{TP_i+FP_i+FN_i+TN_i}}{K} \tag{5}$$

$$\text{Weighted Average Precession (W.A.P)} = \frac{\sum_{i=1}^{K} |S_i| \frac{TP_i}{TP_i+FP_i}}{\sum_{i=1}^{k} |S_i|} \tag{6}$$

$$\text{Weighted Average Recall (W.A.R)} = \frac{\sum_{i=1}^{K} |S_i| \frac{TP_i}{TP_i+FN_i}}{\sum_{i=1}^{k} |S_i|} \tag{7}$$

$$\text{Weighted Average F1\_Score (W.A.F1)} = \frac{\sum_{i=1}^{K} |S_i| \frac{2TP_i}{2TP_i+FP_i+FN_i}}{\sum_{i=1}^{k} |S_i|} \tag{8}$$

		PREDICTED CLASSES				
ACTUAL CLASSES		C_1	C2	…	C_K	
	C_1	**TP_1**				AVC_1
	C_2		**TP_2**			AVC_1
	…	…	…	**…**	…	
	C_K				**TP_K**	AVC_1
		PVC_1	PVC_2		PVC_K	

Figure 11. Confusion matrix of multi-class

Experimental setup and hyper-parameter setting

The present experiment was executed on Jupiter notebook support to implement ML and DL algorithms in Python 3.6.7. The hardware configuration is given in Table 2. The experiment used transfer learning on VGG16 for extracting features and multi-layer perception for classification. Table 2 describes the hardware and software configuration used to train a proposed model for tomato disease classification.

Table 2
Machine specifications

S. No	Hardware and software	Characteristics
1	Memory (RAM)	16 GB
2	Processor	Intel(R) Core i7-10875H CPU@ 2.30 GHz
3	Graphics (GPU)	NVIDIA GeForce RTX 2070-8GB
4	Operating system	Windows 10 and 64 bits
5	Integrated development environment (IDE)	Jupiter Notebook

Hyperparameters are values determined during an algorithm learning process that were optimized to improve the model results. The early stopping condition and dropout ratio were applied for activation, and the nodes of the hidden layers were deactivated while model training to address the model overfitting problem. The hyperparameters of VGG16 selected in the classification layer were Relu, softmax activation functions in hidden layers, and the output layer. SGD (Stochastic Gradient Descent) optimizer, along with learning rate, was 0.0001, dropout was 0.5, decay was 1e-6, momentum was 0.9, patience was 30, the minimum delta was 0.0001, batch size was 8 and epochs were 97. The selected hyperparameters of MLA, such as SVC, were C=10, gamma=0.0001and kernel='rfb'.

VGG16 has three fully connected layers. Among these, two dense layers were fully connected and comprised 4096 nodes. The third layer consisted of 10 neurons corresponding to the number of classes of the dataset. The final layer was the soft-max layer. Hyperparameters, such as activation functions like 'relu' and 'softmax,' dropout, learning rate, decay, momentum, optimizer, batch size, and epochs, are shown in Table 3, and hyperparameters of MLA are shown in Table 4.

Table 3

Hyperparameters of DL

S. No	Hyperparameter	Values
1	Activation functions	Relu, softmax
2	Optimizers	SGD, Adam
3	Learning rate	0.1,0.001,0.0001,0.00001
4	Dropout	0.2,0.3,0.4,0.5
5	Decay	1e-3, 1e-4, 1e-5, 1e-6
6	Momentum	0.8,0.9
7	Patience	15,20,30
8	Minimum delta	0.01, 0.001, 0.0001
9	Batch size	8,16,32,64,128,256
10	Epochs	100, 500, 1000

Table 4

Hyper-parameters of machine learning algorithms

S. No	Machine learning models	Hyperparameter	Values
1	ABC	Learning rate	[0.01, 0.001, 0.001, 0.0001]
		n_estimators	[300, 500, 700, 900]
2	DTC	Criterion	[Gini, entropy]
		max_depth	Range (1, 10)
		min_samples_leaf	Range (1, 5)
		min_samples_split	Range (1, 10)
3	KNN	Metric	[‘minkowski’, ‘euclidean’, ‘manhattan’]
		n_neighbors	[5, 7, 9, 11, 13, 15]
		Weights	[‘uniform’, ‘distance’]
4	LR	C	[100, 10, 1.0, 0.1, 0.01]
		max_iter	[100, 500, 700, 900]
		Penalty	[l1, l2]
		solver	[‘newton-cg’, ‘lbfgs’, ‘liblinear’]

Table 4 *(Continue)*

S. No	Machine learning models	Hyperparameter	Values
5	RFC	Criterion	[Gini, entropy]
		Max_depth	[4, 5, 6, 7, 8]
		max_features	[‘auto’, ‘log2’, ‘sqrt’, 0.33]
		min_samples_leaf	Range (1,5)
		min_samples_splt	Range (1,10)
		n_estimators	[200, 500,700,900]
6	SVC	C	[0.1, 1, 10, 100, 1000]
		Gamma	[1, 0.1, 0.01, 0.001, 0.0001]
		Kernel	[‘rbf’]
7	XGB	Learning rate	[0.01, 0.05, 0.1]
		Max_depth	[3, 5, 7, 9]
		gamma	[0, 0.1, 0.001]

Analysis of Experimental Results

The experimental results were analyzed in five steps. In the first step, the model trained the three fully connected layers of VGG16, MLP, and MLA on dimension reduction level 1 of 10783 non-correlated features. In the second step, the model trained the three fully connected layers of VGG16, MLP, and MLA on dimension reduction level 2 of 5720 principal components. In the third step, the model trained the three fully connected layers of VGG16, MLP, and MLA on dimension reduction level 3 of 1145 optimal features. In the fourth step, the results among the above three-dimension reduction levels were compared, and the best level was selected. In the fifth step, the proposed work results were compared to the previous study on the same dataset.

From the first multi-level dimension reduction, the results were obtained from three fully connected layers of VGG16, MLP, and MLA of 10783 non-correlated features, 5720 principal components, and 1145 optimal features, respectively.

Results of 10783 Non-Correlated Features

The three fully connected VGG16, MLP, and MLA layers on non-correlated features at dimension reduction level 1 are shown in Tables 5, 6, and 7, respectively.

Table 5

Results of VGG16 on 10783 non-correlated features

T.T (H:M:S)	T.A (%)	T.L	V.A (%)	V.L	V.S (903)	
					CP	WP
0:23:55	99.94	0.006	94.91	0.18	857	46

W.A.P (%)	W.A.R (%)	W.A.F1 (%)	S.S	T.P	Tr.P	N.T.P
95.04	94.91	94.88	465 MB	60,993,546	60,993,546	0

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Table 6

Results of MLP on 10783 optimal features

T.T (H:M:S)	T.A (%)	T.L	V.A (%)	V.L	V.S (903)	
					CP	WP
0:14:49	100.0	0.001	94.80	0.17	856	47

W.A.P (%)	W.A.R (%)	W.A.F1 (%)	S.S	T.P	Tr.P	N.T. P
94.93	94.80	94.80	88.3 MB	11,572,746	11,572,746	0

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Table 7

Results of MLA on 10783 optimal features

Classification model	Training time (H: M:S)	Training accuracy (%)	validation accuracy (%)	Total validation samples (903)		W.A.P (%)	W.A.R (%)	W.A.F1 (%)	Storage space
				CP	WP				
LR	0:03:43	98.81	94.35	852	51	94.41	94.35	94.32	843 KB
RFC	2:20:09	100.0	82.17	742	161	82.38	82.17	80.52	107 MB
DTC	0:03:10	59.77	54.93	496	407	50.62	54.93	51.60	18.1 KB

Table 7 *(Continue)*

Classification model	Training time (H: M:S)	Training accuracy (%)	validation accuracy (%)	Total validation samples (903)		W.A.P (%)	W.A.R (%)	W.A.F1 (%)	Storage space
				CP	WP				
ABC	0:35:59	40.80	40.53	366	537	42.71	40.53	30.81	173 KB
KNN	0:00:37	100.0	84.50	763	140	85.82	84.50	83.57	2.05 GB
SVC	0:42:40	99.16	95.13	859	44	95.23	95.13	95.12	328 MB
XGB	0:41:58	100.0	92.13	832	71	92.08	92.14	92.03	3.71 MB

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Results of 5720 Principal Components

The three fully connected layers of VGG16, MLP, and MLA were applied on non-correlated features at dimension reduction level 2, and the results are shown in Tables 8, 9, and 10, respectively.

Table 8

Results of VGG16 on 5720 principal components

T.T (H:M:S)	T.A (%)	T.L	V.A (%)	V.L	V.S (903)	
					CP	WP
0:30:04	100.0	0.006	95.13	0.15	859	44

W.A.P (%)	W.A.R (%)	W.A.F1 (%)	S.S	T.P	Tr.P	N.T.P
95.22	95.13	95.11	307 MB	40,255,498	40,255,498	0

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Table 9

Results of MLP on 5720 principal components

T.T (H: M:S)	T.A (%)	T.L	V.A (%)	V.L	V.S (903)	
					CP	WP
0:04:06	99.99	0.002	95.24	0.18	860	43

Table 9 *(Continue)*

W.A.P (%)	W.A.R (%)	W.A.F1 (%)	S.S	T.P	Tr.P	N.T.P
95.33	95.24	95.22	60.8 MB	7,967,754	7,967,754	0

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Table 10

Results of MLA at 5720 components

Classification Model	Train time (H: M:S)	Training accuracy (%)	Validation accuracy (%)	Total validation samples (903)		W.A.P (%)	W.A.R (%)	W.A.F1 (%)	Storage space
				CP	WP				
LR	0:04:11	98.91	93.24	842	61	93.23	93.24	93.18	447 KB
RFC	2:09:01	99.98	75.19	679	224	75.05	75.19	73.26	119 MB
DTC	0:03:12	59.11	59.03	533	370	54.61	59.03	55.65	18.1 KB
ABC	0:35:12	40.94	41.75	377	526	27.79	41.75	30.67	173 KB
KNN	0:00:15	100.0	84.61	764	139	86.06	84.61	93.73	1.08 GB
SVC	0:26:37	99.07	95.13	859	44	95.23	95.13	95.12	329 MB
XGB	0:44:11	100.0	86.05	777	126	85.94	86.05	85.49	4.63 MB

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Results of 1145 Optimal Features

The three fully connected layers of VGG16, MLP, and MLA were applied on non-correlated features at dimension reduction level 2, and the results are shown in Tables 11, 12, and 13, respectively.

Table 11

Results of VGG16 on 1145 optimal features

T.T (H:M:S)	T.A (%)	T.L	V.A (%)	V.L	V.S (903)	
					CP	WP
0:58:37	100.0	0.0016	95.79	0.15	865	38

W.A.P (%)	W.A.R (%)	W.A.F1 (%)	S.S	T.P	Tr.P	N.T. P
95.88	95.79	95.80	164 MB	21,516,298	21,516,298	0

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Table 12

Results of MLP on 1145 optimal features

T.T (H: M:S)	T.A (%)	T.L	V.A (%)	V.L	V.S (903)	
					CP	WP
0:04:59	99.98	0.0067	95.68	0.16	864	39

W.A.P (%)	W.A.R (%)	W.A.F1 (%)	S.S	T.P	Tr.P	N.T.P
95.71	95.68	95.66	115 MB	15,193,098	15,193,098	0

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Table 13

Results of MLA at 1145 optimal features

Classification model	Training time (H:M:S)	Train accuracy (%)	Validation accuracy (%)	Total validation samples (903)		W.A.P (%)	W.A.R (%)	W.A.F1 (%)	Storage space
				CP	WP				
LR	0:00:46	96.43	92.91	839	64	92.86	92.91	92.83	90.3 KB
RFC	0:25:29	99.96	77.30	698	205	78.49	77.30	75.45	125 MB

Table 13 *(Continue)*

Classification model	Training time (H:M:S)	Train accuracy (%)	Validation accuracy (%)	Total validation samples (903)		W.A.P (%)	W.A.R (%)	W.A.F1 (%)	Storage space
				CP	WP				
DTC	0:00:36	59.11	59.02	533	370	54.59	59.03	55.64	18.1 KB
ABC	0:06:38	40.95	41.75	377	526	27.79	41.75	30.67	173 KB
KNN	0:00:02	100.0	85.94	776	127	87.09	85.94	85.19	223 MB
SVC	0:02:51	97.97	94.35	852	51	94.44	94.35	94.35	60.9 MB
XGB	0:08:43	100.0	87.04	786	87	86.80	87.04	86.59	4.74 MB

Note. T.T-Training time; T.A-Training accuracy; T.L-Train loss; V.A-Validation accuracy; V.L-Validation loss; V.S-Validation samples; CP-Correct predictions; WP-Wrong predictions; W.A.P-Weighted average precession, W.A.R- Weighted average recall; W.A.F1_score- Weighted average F1_Score; S.S-Storage space; T.P-Total parameters; Tr. P-Trainable parameters; N.T.P- Non-trainable parameters

Tables 14 and 15 and Figure 12 show the comparison between the without and with dimension reduction results and the optimal results of the highest validation accuracy of 95.79% and weighted average F1_Score of 95.80% obtained at level 3 of dimension reduction by three fully connected (fc) layers of VGG16. Training and validation loss curves of 3fc of VGG16 and MLP without and with dimensional reduction levels are shown in Figures 13(a) to (d). The lowest validation loss was identified at level 3 by 3fc of VGG16 in Figure 13(d). Similarly, the training and validation accuracy curves of 3fc of VGG16 and MLP without and with dimension reduction levers are shown in Figures 14(a) to (d). The highest validation accuracy of 95.79% was identified at level 3 by 3fc of VGG16, as shown in Figure 14(d). The confusion matrix of 3fc of VGG16, SVC, and MLP without and with dimension reduction levels are shown in Figures 15(a) to (e). Of the 906 validation samples of images (0.05%), 865 correct predictions (CP) and 38 wrongly predicted (WP) samples are identified in level 3 by 3fc of VGG16 and shown in Figure 15(e). The classification reports of 3fc of VGG16, SVC, and MLP without and with dimension reduction levels are shown in Figures 16(a) to 16(e). The optimal results of VGG16 are precession, recall, and f1_scores are shown in Figure 16(e).

Table 14

Comparing the performance measure of classification models at three different dimension reduction levels, including without dimension reduction

Dimension reduction level	Features/ components	Model	Training accuracy	Training loss	Validation accuracy	Validation loss	Weighted average F1_score
0	25088	VGG16	99.91	0.0125	94.80	0.18	94.82
1	10783	SVC	99.16	-	95.13	-	95.12
2	5720	MLP	99.99	0.002	95.24	0.17	95.22
3	1145	MLP	99.98	0.0067	95.68	0.16	95.66
3	1145	VGG16	100.0	0.0016	95.79	0.15	95.80

Table 15

Comparing the parameters, training time, and storage space of classification models at three different dimension reduction levels, including without dimension reduction

Dimension reduction level	Validation samples-903 (0.05%)		Training time (HH:MM:SS)	Total parameters	Trainable parameters	Non-trainable parameters	Storage Space (MB)
	CP	WP					
0	856	47	23:30:33	134,301,514	119,586,826	14,714,688	968
1	859	44	0:42:40	-	-	-	626
2	860	43	0:04:06	21,967,754	21,967,754	0	260
3	864	39	0:04:59	15,193,098	15,193,098	0	115
3	865	38	0:58:37	21,516,298	21,516,298	0	164

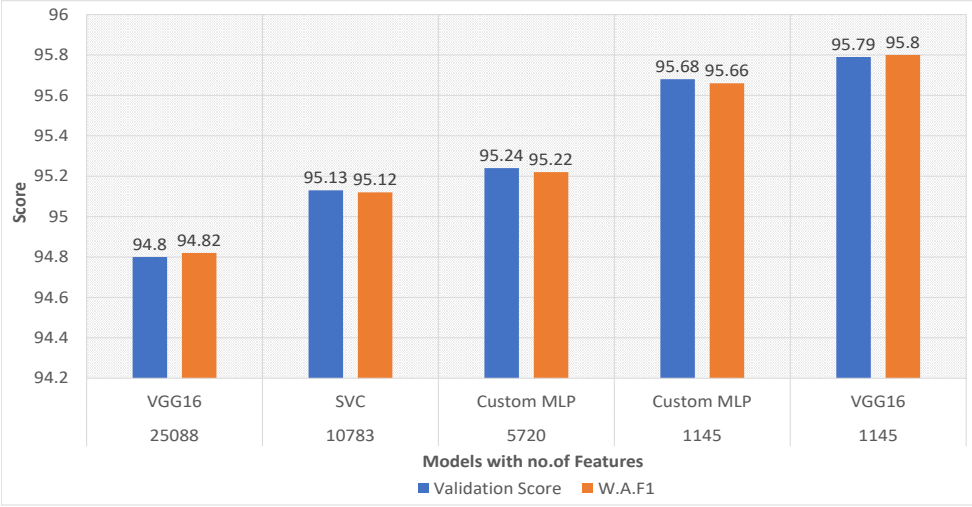
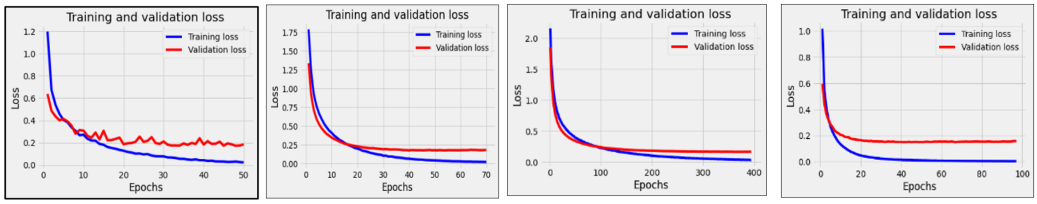
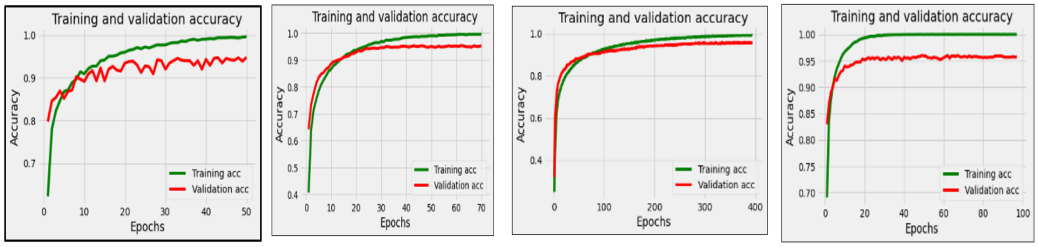


Figure 12. Comparison of results at three different dimension reduction levels



(a) (b) (c) (d)

Figure 13. Training and validation loss: (a) without dimensionality reduction; (b) at principal components 5720 by MLP; (c) at optimal features 1145 by MLP; (d) at optimal features 1145 VGG16



(a) (b) (c) (d)

Figure 14. Training and validation accuracy: (a) without dimensionality reduction; (b) at principal components 5720 by MLP; (c) at optimal features 1145 by MLP; (d) at optimal features 1145 by VGG16

Confusion Matrix of VGG16 at extracted features 25088 (Without dimension reduction)

TRUE LABELS	T. B	T. E	T. L	T. Le	T. Se	T. Sp	T. T	T.T. Y	T.M	T. H
T. B	414	7	0	0	0	0	0	4	0	0
T. E	3	187	2	2	0	4	0	2	0	0
T. L	2	187	2	2	0	4	0	2	0	0
T. Le	2	123	239	10	0	3	1	2	0	1
T. Se	43	106	1	14	178	3	2	2	3	2
T. Sp	0	13	0	0	0	318	1	3	0	0
T. T	9	77	0	1	0	45	141	3	0	4
T.T. Y	6	16	0	0	0	2	0	1047	0	0
T.M	0	3	0	9	1	3	0	0	58	0
T. H	2	3	0	0	0	3	0	0	0	310

PREDICTED LABELS

(a)

Confusion Matrix of SVC at Non-correlated feature 10783

TRUE LABELS	T. B	T. E	T. L	T. Le	T. Se	T. Sp	T. T	T.T. Y	T.M	T. H
T. B	106	0	0	0	0	0	0	0	0	0
T. E	1	38	3	2	0	0	4	2	0	0
T. L	0	9	83	3	0	0	0	1	0	0
T. Le	0	0	0	46	0	0	0	0	0	0
T. Se	0	1	2	4	80	0	0	1	0	0
T. Sp	0	1	0	0	0	79	3	0	0	0
T. T	0	0	1	0	0	4	64	0	0	0
T.T. Y	0	0	0	0	0	1	0	266	0	0
T.M	0	0	0	0	0	0	0	0	18	0
T. H	0	0	0	0	0	0	0	0	0	79

PREDICTED LABELS

(b)

Confusion Matrix of MLP at Principal components 5720

TRUE LABELS	T. B	T. E	T. L	T. Le	T. Se	T. Sp	T. T	T.T. Y	T.M	T. H
T. B	106	0	0	0	0	0	0	0	0	0
T. E	1	38	1	2	0	0	6	2	0	0
T. L	0	5	85	2	1	0	1	0	0	1
T. Le	0	0	0	46	0	1	0	0	0	0
T. Se	0	2	1	3	81	0	0	1	0	0
T. Sp	0	1	0	0	0	78	4	0	0	0
T. T	0	1	0	0	1	3	63	0	0	2
T.T. Y	0	0	0	1	0	0	0	266	0	0
T.M	0	0	0	0	0	0	0	0	18	0
T. H	0	0	0	0	0	0	0	0	0	79

PREDICTED LABELS

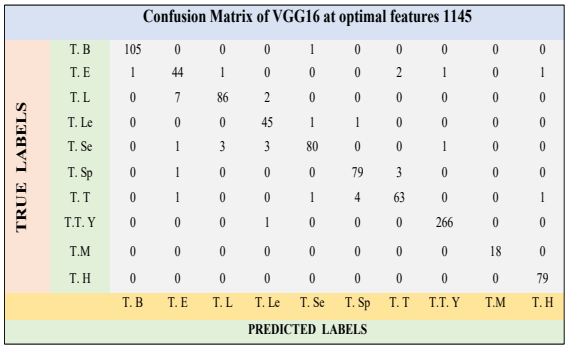
(c)

Confusion Matrix of MLP at optimal features 1145

TRUE LABELS	T. B	T. E	T. L	T. Le	T. Se	T. Sp	T. T	T.T. Y	T.M	T. H
T. B	105	0	0	0	1	0	0	0	0	0
T. E	1	42	2	0	1	0	2	1	0	1
T. L	0	4	86	3	0	0	1	0	0	1
T. Le	0	0	0	45	1	1	0	0	0	0
T. Se	0	1	1	3	81	0	0	1	1	0
T. Sp	0	1	0	0	0	78	3	0	0	1
T. T	0	0	0	0	1	3	63	0	0	3
T.T. Y	0	0	0	0	0	0	0	267	0	0
T.M	0	0	0	0	0	0	0	0	18	0
T. H	0	0	0	0	0	0	0	0	0	79

PREDICTED LABELS

(d)



(e)

Figure 15. Confusion matrices: (a) VGG16 without dimension reduction; (b) SVC at non correlated feature 10783; (c) MLP at principal components 5720; (d) MLP at optimal features 1145; (e) VGG16 at optimal features 1145.

Note. **T.B**=Tomato Bacterial spot; **T.E**=Tomato Early blight; **T.L**=Tomato Late blight; T**.Le**=Tomato_ Leaf Mold; **T.Se**=Tomato_Septoria_leaf_spot; **T.Sp**=Tomato_Spider_mites Two-spotted_spider_mite; **T.T**=Tomato_Target Spot; **T.T.Y**=Tomato Tomato Yellow_Leaf_Curl_Virus; T**.M**=Tomato Tomato mosaic virus; **T.H**=Tomato_healthy

Classes	Precision	Recall	F1_Score	Support
T. B	0.9906	0.9906	0.9906	106
T. E	0.7407	0.8000	0.7692	50
T. L	0.9630	0.8211	0.8864	95
T. Le	0.7966	1.0000	0.8868	47
T. Se	0.9643	0.9205	0.9419	88
T. Sp	0.9412	0.9639	0.9524	83
T. T	0.9130	0.9000	0.9065	70
T.T. Y	0.9888	0.9963	0.9925	267
T.M	1.0000	1.0000	1.0000	18
T. H	1.0000	0.9873	0.9936	79
Accuracy			**0.9480**	903
Macro AVG	0.9298	0.9380	0.9320	903
Weighted AVG	0.9511	0.9480	0.9482	903

(a)

Classes	Precision	Recall	F1_Score	Support
T. B	0.9907	1.0000	0.9953	106
T. E	0.7755	0.7600	0.7677	50
T. L	0.9326	0.8737	0.9022	95
T. Le	0.8364	0.9787	0.9020	47
T. Se	1.0000	0.9091	0.9524	88
T. Sp	0.9405	0.9518	0.9461	83
T. T	0.9014	0.9143	0.9078	70
T.T. Y	0.9815	0.9963	0.9888	267
T.M	1.0000	1.0000	1.0000	18
T. H	1.0000	1.0000	1.0000	79
Accuracy			**0.9513**	903
Macro AVG	0.9359	0.9384	0.9362	903
Weighted AVG	0.9523	0.9513	0.9512	903

(b)

Classes	Precision	Recall	F1_Score	Support
T. B	0.9907	1.0000	0.9953	106
T. E	0.8085	0.7600	0.7835	50
T. L	0.9770	0.8947	0.9341	95
T. Le	0.8519	0.9787	0.9109	47
T. Se	0.9759	0.9202	0.9474	88
T. Sp	0.9512	0.9398	0.9455	83
T. T	0.8514	0.9000	0.8750	70
T.T. Y	0.9888	0.9963	0.9925	267
T.M	1.0000	1.0000	1.0000	18
T. H	0.9634	1.0000	0.9814	79
Accuracy			**0.9524**	903
Macro AVG	0.9359	0.9390	0.9365	903
Weighted AVG	0.9533	0.9524	0.9522	903

(c)

Labels	Precision	Recall	F1_Score	Support
T. B	0.9906	0.9906	0.9906	106
T. E	0.8750	0.8400	0.8571	50
T. L	0.9663	0.9053	0.9348	95
T. Le	0.8824	0.9574	0.9184	47
T. Se	0.9529	0.9205	0.9364	88
T. Sp	0.9512	0.9398	0.9455	83
T. T	0.9130	0.9000	0.9065	70
T.T. Y	0.9926	1.0000	0.9963	267
T.M	0.9474	1.0000	0.9730	18
T. H	0.9294	1.0000	0.9634	79
Accuracy			**0.9568**	903
Macro AVG	0.9401	0.9453	0.9422	903
Weighted AVG	0.9571	0.9568	0.9566	903

(d)

Classes	Precision	Recall	F1_Score	Support
T. B	0.9906	0.9906	0.9906	106
T. E	0.8148	0.8800	0.8462	50
T. L	0.9556	0.9053	0.9297	95
T. Le	0.8824	0.9574	0.9184	47
T. Se	0.9639	0.9091	0.9357	88
T. Sp	0.9405	0.9518	0.9461	83
T. T	0.9265	0.9000	0.9130	70
T.T. Y	0.9925	0.9963	0.9944	267
T.M	1.0000	1.0000	1.0000	18
T. H	0.9753	1.0000	0.9875	79
Accuracy			**0.9579**	903
Macro AVG	0.9442	0.9490	0.9462	903
Weighted AVG	0.9588	0.9579	0.9580	903

(e)

Figure 16. Classification reports: (a) VGG16 without dimension reduction; (b) SVC at non correlated feature 10783; (c) MLP at principal components 5720; (d) MLP at optimal features 1145; (e) VGG16 at optimal features 1145

Note. **T.B**=Tomato Bacterial spot; **T.E**=Tomato Early blight; **T.L**=Tomato Late blight; T**.Le**=Tomato_ Leaf Mold; **T.Se**=Tomato_Septoria_leaf_spot; **T.Sp**=Tomato_Spider_mites Two-spotted_spider_mite; **T.T**=Tomato_Target Spot; **T.T.Y**=Tomato Tomato Yellow_Leaf_Curl_Virus; T**.M**=Tomato Tomato mosaic virus; **T.H**=Tomato_healthy

Table 16

Comparison of the performance measures of proposed models with previous models on 18169 images of the tomato dataset

Methods	Training accuracy (%)	Train loss	Validation accuracy (%)	Validation loss	Weighted average F1_score (%)
DNN (Gadekallu et al., 2021)	99	NA	94	NA	NA
Variation of the LeNet (Tm et al., 2018)	99.3	NA	94.8	NA	94.81
Squeeze Net (Durmus et al., 2017)	NA	NA	94.3	NA	NA
AlexNet (Durmus et al., 2017)	NA	NA	95.65	NA	NA
Filter+PCA+Boruta (proposed)+MLP	99.98	0.0067	95.68	0.16	95.66
Filter+PCA+Boruta (proposed)+VGG16	100.0	0.0016	95.79	0.15	95.80

Table 17

Comparison of the parameters, training time, and storage space of proposed models to previous models on 18169 images of the tomato dataset

Methods	Total parameters	Trainable parameters	Non-Trainable parameters	Train Time (HH:MM:SS)	Storage Space (MB)
DNN (Gadekallu et al., 2021)	NA	NA	NA	NA	NA
variation of the LeNet (Tm et al., 2018)	NA	NA	NA	NA	NA
Squeeze Net (Durmus et al., 2017)	NA	NA	NA	NA	2.9
AlexNet (Durmus et al. 2017)	NA	NA	NA	NA	227.6
Filter+PCA+Boruta (proposed)+MLP	15,193,098	15,193,098	0	0:04:59	115
Filter+PCA+Boruta (proposed)+VGG16	21,516,298	21,516,298	0	0:58:37	164

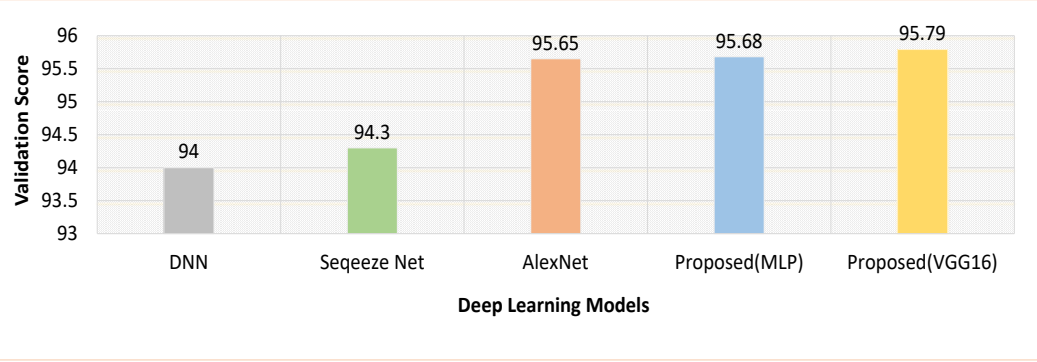


Figure 17. Comparison of the validation accuracy of the proposed models with previous models

Comparison of the Validation Accuracy of the Proposed Model with Previous Models

The performance of the proposed models was compared to the previously proposed DL models: DNN, SeqeezeNet, and AlexNet. Figure 17 shows the comparison of the proposed model to previous models on tomato images of the Plant Village dataset. Tables 16 and 17 show the comparison output of proposed models with previously developed models of train accuracy, train loss, validation accuracy, validation loss, weighted average f1_score, trainable parameters, non-trainable parameters, and storage space. The highest validation

accuracy of 95.68% and 95.79% and the highest weighted average F1_score of 95.66% and 95.80% were obtained in the proposed multi-level dimension algorithm by MLP and 3fc of VGG16, respectively. The validation accuracy of the proposed models was improved compared to that of the previous models, such as DNN (94%), SqueezeNet (94.3%), and AlexNet (95.65%).

CONCLUSION

In the present study, multi-level dimensionality reduction-based algorithms, Filter and PCA, and Boruta feature methods were developed to obtain optimal features. The classification performance of VGG16, MLP, and MLA was compared at each level of optimal features. Finally, it was concluded that level 3 of dimension reduction provides 1145 optimal features, recorded as the best among all the previous studies. MLP and VGG16 provided the best validation accuracy of 95.68% and 95.79%, respectively.

FUTURE RESEARCH DIRECTION

The present study examined the prediction of tomato leaf diseases based on the proposed multi-level dimension reduction algorithm. Therefore, in future studies, a robust model may be developed by adding other dimension reduction techniques, such as particle swarm optimization (PSO), linear discernment analysis (LDA), and autoencoders to overcome the overfitting problem and identify the diseases in plants, crops, and vegetables based on the images and prevent crop loss at an early stage in favor of farmers. A minor limitation in the proposed approach is that the Filter method for identifying non-correlated features and the Boruta feature selection algorithm for obtaining the optimal features are time-consuming processes.

ACKNOWLEDGEMENT

Pondicherry University, Puducherry, under the guidelines of research supervisor Dr. Ramasami Lakshmi, supported this research. We (Premkumar Borugadda, Ramasami Lakshmi, and Satyasangram Sahoo), the authors of the research paper titled “Transfer Learning VGG16 Model for Classification of Tomato Plant Leaf Diseases: A Novel Approach for Multi-level Dimensional Reduction,” would like to express our deep sense of gratitude for the entire editorial team of Pertanika Journal of Science and Technology for accepting the manuscript.

REFERENCES

Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. *International Journal of Computer Science, 9*(5), Article 272.

Awad, M., & Khanna, R. (2015). Support vector machines for classification. In *Efficient Learning Machines* (pp. 39-66). Apress Berkeley, CA. https://doi.org/10.1007/978-1-4302-5990-9_3

Behera, B., Kumaravelan, G., & Kumar, P. (2019). Performance evaluation of deep learning algorithms in biomedical document classification. In *2019 11th International Conference on Advanced Computing (ICoAC)* (pp. 220-224). IEEE Publishing. https://doi.org/10.1109/ICoAC48765.2019.246843

Cerda, P., & Varoquaux, G. (2020). Encoding high-cardinality string categorical variables. *IEEE Transactions on Knowledge and Data Engineering, 34*(3), 1164-1176. https://doi.org/10.1109/TKDE.2020.2992529

Chen, G., & Chen, J. (2015). A novel wrapper method for feature selection and its applications. *Neurocomputing, 159*, 219-226. https://dl.acm.org/doi/abs/10.5555/2781902.2782171

Chuanlei, Z., Shanwen, Z., Jucheng, Y., Yancui, S., & Jia, C. (2017). Apple leaf disease identification using genetic algorithm and correlation-based feature selection method. *International Journal of Agricultural and Biological Engineering, 10*(2), 74-83. https://doi.org/10.3965/j.ijabe.20171002.2166

Doquire, G., & Verleysen, M. (2013). A graph Laplacian based approach to semi-supervised feature selection for regression problems. *Neurocomputing, 121*, 5-13. https://doi/abs/10.1016/j.neucom.2012.10.028

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. *Journal of Biomedical Informatics, 35*(5-6), 352-359. https://doi.org/10.1016/S1532-0464(03)00034-0

Durmuş, H., Güneş, E. O., & Kırcı, M. (2017). Disease detection on the leaves of the tomato plants by using deep learning. In *2017 6th International Conference on Agro-Geoinformatics* (pp. 1-5). IEEE Publishing. https://10.1109/Agro-Geoinformatics.2017.8047016

Gadekallu, T. R., Rajput, D. S., Reddy, M., Lakshmanna, K., Bhattacharya, S., Singh, S., & Alazab, M. (2021). A novel PCA-whale optimization-based deep neural network model for classification of tomato plant diseases using GPU. *Journal of Real-Time Image Processing, 18*(4), 1383-1396. https://doi.org/10.1007/s11554-020-00987-8

Gao, B., & Pavel, L. (2017). *On the properties of the softmax function with application in game theory and reinforcement learning*. arXiv Preprint. https://doi.org/10.48550/arXiv.1704.00805

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In R. Meersman, Z. Tari & D. C. Schmidt (Eds.), *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (pp. 986-996). Springer. https://doi.org/10.1007/978-3-540-39964-3_62

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). *Squeeze Net: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size*. arXiv Preprint. https://doi.org/10.48550/arXiv.1602.07360

Khammari, A., Nashashibi, F., Abramson, Y., & Laurgeau, C. (2005). Vehicle detection combining gradient analysis and AdaBoost classification. In *Proceedings. 2005 IEEE Intelligent Transportation Systems* (pp. 66-71). IEEE Publishing. https:// doi:10.1109/ITSC.2005.1520202

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25* (p. 1). Morgan Kaufmann Publishers.

Kursa, M. B., & Rudnicki, W. R. (2010). Feature selection with the Boruta package. *Journal of Statistical Software, 36,* 1-13. https://doi.org/10.18637/jss.v036.i11

Li, J., Si, Y., Xu, T., & Jiang, S. (2018). Deep convolutional neural network-based ECG classification system using information fusion and one-hot encoding techniques. *Mathematical Problems in Engineering, 2018*, Article 7354081. https://doi.org/10.1155/2018/7354081

Ma, H., Li, Y., Chen, Q., Zhang, L., & Xu, J. (2018). A single-stage integrated boost-LLC AC–DC converter with quasi-constant bus voltage for multichannel LED street-lighting applications. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, *6*(3), 1143-1153. https://doi.org/10.1109/JESTPE.2018.2847327

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science, 7*, Article 1419. https://doi.org/10.3389/fpls.2016.01419

Mudrova, M., & Procházka, A. (2005, November 15). Principal component analysis in image processing. In *Proceedings of the MATLAB Technical Computing Conference* (pp. 1-4). Prague, Czech Republic.

Noriega, L. (2005). *Multilayer perceptron tutorial*. Staffordshire University. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=4c8339b893423f1e14e34cc1543faee4e5ee4244

Ramchoun, H., Ghanou, Y., Ettaouil, M., & Idrissi, M. A. J. (2016). Multilayer perceptron: Architecture optimization and training. *International Journal of Interactive Multimedia and Artificial Intelligence, 4*(1), 26-30. http://doi.org/10.9781/ijimai.2016.415

Simonyan, K., & Zisserman, A. (2014). *Very deep convolutional networks for large-scale image recognition*. arXiv Preprint. https://doi.org/10.48550/arXiv.1409.1556

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management, 45*(4), 427-437. https://doi.org/10.1016/j.ipm.2009.03.002

Tammina, S. (2019). Transfer learning using VGG-16 with deep convolutional neural network for classifying images. *International Journal of Scientific and Research Publications, 9*(10), 143-150. https://doi.org/10.29322/IJSRP.9.10.2019.p9420

Tang, Y., & Wu, X. (2016). Saliency detection via combining region-level and pixel-level predictions with CNNs. In *European Conference on Computer Vision* (pp. 809-825). Springer. https://doi.org/10.48550/arXiv.1608.05186

Tm, P., Pranathi, A., SaiAshritha, K., Chittaragi, N. B., & Koolagudi, S. G. (2018). Tomato leaf disease detection using convolutional neural networks. In *2018 eleventh international conference on contemporary computing (IC3)* (pp. 1-5). IEEE Publishing. https://doi.org/10.1109/IC3.2018.8530532

Torlay, L., Perrone-Bertolotti, M., Thomas, E., & Baciu, M. (2017). Machine learning–XGBoost analysis of language networks to classify patients with epilepsy. *Brain Informatics, 4*(3), 159-169. https://doi.org/10.1007/s40708-017-0065-7

Zhang, M. L., & Zhou, Z. H. (2005). A k-nearest neighbor-based algorithm for multi-label classification. In *2005 IEEE International Conference on Granular Computing* (Vol. 2, pp. 718-721). IEEE Publishing. https://doi.org/10.1109/GRC.2005.1547385