SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V10I8P104

Volume 10 Issue 8, 37-50, August 2023
© 2023 Seventh Sense Research Group®

Original Article

A Hybrid Model using MobileNetv2 and SVM for Enhanced Classification and Prediction of Tomato Leaf Diseases

N. F. Esomonu[1], U. F. Eze[2], A. M. John-Otumu[3], I. I. Ayogu[4], O. C. Nwokonkwo[5], E. O. Oshoiribhor[6], S. A. Okolie[7], E. C. Nwokorie[8], C. V. Mbamala[9], O. Dokun[10]

[1,2,3,4,5,9,10]*Department of Information Technology, Federal University of Technology, Owerri, Nigeria.*
[6]*Department of Computer Science, Ambrose Alli University, Ekpoma, Nigeria.*
[7,8]*Department of Computer Science, Federal University of Technology, Owerri, Nigeria.*

[3]*Corresponding Author : macgregor.otumu@gmail.com*

Received: 04 June 2023 Revised: 08 July 2023 Accepted: 06 August 2023 Published: 31 August 2023

Abstract - *This study proposes an innovative method for automated categorising tomato leaf diseases using a hybrid model that combines deep learning and machine learning approaches. The proposed model integrates Convolutional Neural Networks (CNNs) pre-trained model (MobileNetv2) for feature mining and SVM for disease multi-class classification and prediction. A comprehensive dataset of 10,000 tomato leaf images was collected, preprocessed, and utilized for model training and evaluation. Experimental results demonstrate the efficacy of the hybrid model, achieving an impressive accuracy ranging from 88% to 99% in multi-class disease image classification. According to the comparative analysis, the relevant models' accuracies range from 70% to 97%. The outcome further endorses the proposed techniques' superiority. It highlights the necessity of choosing the appropriate model architecture and optimization strategies for obtaining high accuracy in image classification tasks, indicating its potential for real-world deployment. This study adds to the advancement of agronomic know-how by providing an efficient and accurate tool for tomato disease management, enabling early detection, and supporting sustainable crop production.*

Keywords - *Deep learning, Pre-trained models, Machine learning, Classification, Prognosis, Tomato diseases.*

1. Introduction

Agriculture is the mother of nature and a significant resource needed for the growth of any economy [1]. Innovative technologies in agriculture, such as weather forecasting, disease prediction, and crop harvesting, can improve the quality and quantity of agricultural products [2]. The world's population has been steadily increasing, reaching 7 billion in 2017, and is projected to reach 9.7 billion by 2050 and 10.9 billion by 2100 [3, 4]. The demand for agricultural products has risen due to rapid population growth, leading to an expansion of cultivation [5]. To provide for the rising demand for food, it is essential to double crop yield production by 2050 with a yearly improvement of 2.4% [6].

Nigeria's agricultural sector contributes about 23% to the Gross Domestic Product (GDP) [7]. Small-scale farming is the predominant practice in Nigeria, but it faces challenges in meeting the food demand and ensuring food security [8]. Tomatoes are a common and extensively used vegetable crop in Nigeria and globally [9].

Tomato cultivation has health benefits, generates revenue, and creates employment for rural and urban populations [10]. Tomato plants belong to the Solanaceae family and have various cultivars. The fruit is the most critical component of the tomato plant, and it can be consumed fresh or processed into various products [11].

Pests and diseases threaten tomato production, resulting in substantial losses [12, 13]. Disease outbreaks, such as the Tuta absoluta pest, have caused significant damage to tomato crops in Nigeria, leading to high losses in production [7]. Early detection of plant diseases is crucial for effectively managing and preventing extensive damage [14].

Advancements in technology, particularly the integration of machine and deep learning models, can offer better opportunities for improved disease classification, prediction and management practices [14-16]. These technologies can also be implemented for the early classification and detection of tomato diseases to aid farmers in improving yields and sales.

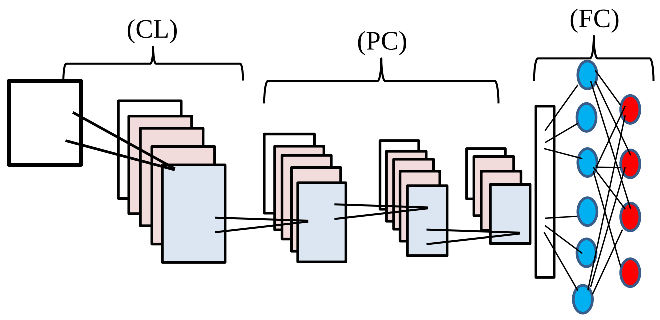


Fig. 1 Traditional architecture for CNN

In recent years, deep learning models have become more popular for image classification and regression issues. Because of their thick layers, these models have shown notable progress in artificial intelligence. In the past five years, deep Convolutional Neural Networks (ConvNets or CNNs) have been essential in developing image-based applications. In contrast to prior methods, CNNs reduce the need for intensive pre-processing of images or data before putting them into the network. They are excellent at automatically extracting features from images, which improves the neural network's capacity for learning. Ultimately, CNNs, in particular, have revolutionized image analysis and excelled in various applications [17].

A Convolution Neural Network (CNN) has several layers that employ a differential function to transform an input volume into an output volume. The visual cortex of the brain influenced the design of CNN. The design more closely resembles the data because fewer parameters are needed, and weights can be reused. Convolution (CONV), Pooling (Pool), and Fully Connected (FC) are just a few of the various layer types that make up a convolutional network.

The CNN layers enable the extraction of convolutional features. The recovered deep characteristics are crucial to the decision support system. The 2D convolution is generated by Equations (1) and (2), respectively.

$$\mathrm{y}[\mathrm{m},\mathrm{n}] = x[m,n]\,.\,h[m,n] \tag{1}$$

$$y[m,n] = \sum_{i=-\propto}^{\propto} \sum_{j=-\propto}^{\propto} x[i,j]\,.\,h[m-i,n-j] \tag{2}$$

Where,

x[m, n]	=	Input
m, n	=	both the number of rows and columns correspondingly
i, j	=	row index and column index

Similarly, Equation (3) provides the size of the image following convolution:

$$size = \left[\left(\frac{m+2p-n}{s}+1,\ \frac{m+2p-n}{s}+1\right)\right] \tag{3}$$

Where,

m	=	quantity of input characteristics; n = size of the convolution kernel
p	=	padding
s	=	stride

Equation (4) is used to calculate the mini-batch mean mathematically.

$$\mu\beta = \frac{1}{m}\sum_{i-1}^{m} x_i \tag{4}$$

The variance in the mini-batch is seen in Equation (5)

$$\sigma_{\beta}^{2} = \frac{1}{m}\sum_{i-1}^{m}(x_i - \mu\beta)^{2} \tag{5}$$

Normalization is provided by Equation (6)

$$\overline{x_i} = \frac{x_i - \mu\beta}{\sqrt{\sigma_{\beta}^{2}+\varepsilon}} \tag{6}$$

Clear-cut cross-entropy is presented in Equation (7) like this:

$$L[y,y] = -\sum_{j=0}^{M}\ \sum_{i=0}^{M}[y_{ij}\ x\ Log(y_{ij})] \tag{7}$$

This research aims to develop an efficient hybrid prognosis technique for early multi-class tomato leaf disease utilizing a lightweight variant architecture of CNN called MobileNetv2 cascaded with SVM. The MobileNetV2 is used to mine meaningful structures from the tomato leaf images, while the SVM is employed for multi-class classification.

2. Related Works

Several research studies have been conducted in classifying and predicting tomato leaf diseases using artificial intelligence techniques. Altunta & Kocamaz [18] constructed a Deep-CNN for forecasting tomato leaves illnesses, using 18,835 datasets from the plant village on Kaggle. They trained the model with 40% of the dataset and reserved 10% for testing. The ResNet50 model achieved an accuracy of 96.99%, with suggestions to combine multiple approaches for improved effectiveness. However, only 50% of the dataset was used, indicating potential overfitting.

In another study by Liu & Wang [19], the authors designed a CNN model for predicting diseases from tomato leaves. The experiment used a specific hardware setup and three pre-trained models: SSD, Faster R-CNN, and YOLO v3. The primary dataset consisted of 1,952 low-quality tomato leaf images. YOLO v3 showed a detection time of 20.39 ms and an accuracy of 92.39%.

In a different investigation by Hiremath and Gowda [20], the authors created a diagnostic model for classifying ten tomato diseases using a CNN-based approach. They used around 5,000 images downloaded from the Kaggle repository dataset and supplemented them where possible. The model realized 91.2% accuracy rate using three pretrained networks: MobileNet, VGG16, and InceptionV3.

Damicone & Brandenberger [21] utilized CNN and RNN techniques to develop a diagnostic model for identifying tomato leaf diseases. The study used 1,500 datasets without specifying their sources. The model attained 86.18% accuracy rate.

Mengistu et al. [22] developed a hybrid model combining CNN and KNN for forecasting tomato diseases. They used 10,000 images from the Kaggle Plant Village dataset. The dataset was splitted into two; 75% for training and 25% for testing. The CNN-based model outperformed the KNN model in terms of accuracy, precision, recall, and f1-score but values were not recorded.

Islam et al. [23] utilized CNN deep learning to predict tomato leaf diseases. They trained the network on a powerful machine with specific hardware specifications and used digital images from the Kaggle Plant Village dataset and the plant factory at Ehime University. The model achieved high accuracy and performance metrics.

Mkonyi et al. [24] developed a deep learning-CNN model for classifying and detecting tomato diseases using Python and Keras. They used a transfer learning approach and a dataset of 2,145 images. The proposed technique achieved a maximum accuracy rate, but the study emphasized the need for more data to improve classification performance.

Gunarathna et al. [25] created a deep learning-CNN model for recognizing tomato diseases. They used a laptop with specific specifications and collected 22,930 datasets from the internet. The proposed CNN model and pre-trained models achieved specific accuracy rates.

Trivedi et al. [26] proposed a CNN model for probing and prognosis of tomato plant diseases experimented over Google Colab. They used a dataset of 3,000 images downloaded from kaggle plant village to train the model using different pre-trained models. The CNN model achieved high accuracy, but the study acknowledged the limitations of the small dataset. Some level of disease management for abiotic diseases was mentioned.

Again, Khatoon et al. [27] also proposed a model based on CNN for predicting 24 diseases in tomato leaves and fruits. They used Python and the Keras library on a GPU-based workstation. Three pre-trained network models were employed, including DenseNet, ResNet, and VGG16. The study used 23,716 images from adjacent farms and public datasets, training (80%) and testing (20%). The accuracy rate for the DenseNet was reported as 95.31%.

Using leaf data, Ramakrishna [28] employed deep and Machine Learning (ML) techniques to predict five tomato plant diseases. They utilized Python 3.7.3 and implemented VGG19, VGG16, and Random Forest XGBoost methods. The study used 6,500 images from the ImageNet dataset, with 20% for testing and 80% for training. Only the accuracy rate of VGG19 96% was reported.

Sreelatha et al. [29] built an ML predictive model using the KNN approach to detect 14 tomato disorders. They collected 1,500 digital images but did not specify the sources. The dataset was divided into testing (20%) and training (80%). The proposed method achieved an accuracy rate of 86.18%, but it was noted that KNN is computationally expensive.

Al-gaashani et al. [30], the authors offered a model for predicting tomato diseases across 38 classes and 14 species using multinomial logistic regression. They used the Keras library and Python on a machine with specific specifications. The study utilized 1,152 images from the kaggle repository dataset, with 75% of the dataset for training and 25% for testing. The multinomial logistic regression achieved a 97% accuracy rate.

Qasrawi et al. [31] conducted an experimental investigation on tomato leaf disease identification in Palestine using six ML algorithms: Neural Networks (NN), SVM, Decision Tree (DT), Logistic Regression (LR), Naive Bayes (NB), and Random Forest (RF). A dataset of 3,000 images was collected, with 30% used for training and 70% for testing. The results showed that logistic regression and neural networks achieved 70% accuracy.

Aversano et al. [32] developed a deep-learning CNN model with ten classes for recognizing diseases in tomato leaves. They utilized pre-trained models such as VGG-19, Xception, and ResNet-50. The dataset consisted of 16,000 images from Plant Village on Kaggle.com. The model achieved a precision of 97%. Nagamani & Sarojadevi [33] and Sheema et al. [34] were presented a hybrid model combining fuzzy SVM and R-CNN for forecasting 75 tomato leaf diseases. They used 735 digital images from the Plant Village dataset. The dataset was split into 70% for training and 30% for testing. The model achieved an accuracy rate of 96.7%.

Gonzalez-Huitron et al. [35] developed a model combining deep learning and machine learning techniques for predicting tomato leaf diseases in 10 classes. They utilized pre-trained models like Xception, MobileNetV2, NasNetMobile, and MobileNetV3. The dataset consisted of 18,215 images from Plant Village on Kaggle.com. The dataset was split into 70% for training and 30% for testing. The model achieved a 94.5% accuracy rate.

Sumalatha et al. [35] also developed a deep-learning CNN model for forecasting ten types of tomato leaf diseases. They used pre-trained models like Xception, VGG16, InceptionV3, MobileNet, ResNet50, and DenseNet121. The dataset included 11,333 digital images of tomato diseases from Plant Village on Kaggle.com. The dataset was split into 10% for testing, 10% for assessment, and 80% for training. The accuracy rate of the DenseNet121 model was 95.48%.

Nithish et al. [37] and Vatsal Mahajan et al.[38] were developed a deep-learning CNN model to recognize ten different tomato leaf disease types. The dataset consisted of 12,206 images from the Plant Village dataset on Kaggle.com. The dataset was split into 20% for testing and 80% for training. The model included the ResNet50 pre-trained model and achieved a 97% accuracy rate.

Hong et al. [39] and Gargi Sharma, and Gourav Shrivastava [40] were developed a deep-learning CNN model to predict tomato leaf disorders in 10 classes. They employed pre-trained models such as ShuffleNet, Xception, Resnet50, MobileNet, and DenseNet121. Their dataset consisted of 41,263 images from Plant Village on Kaggle, with an 80/20 split for training and testing. Despite potential imbalances in the datasets, the model achieved an accuracy of 91.10%.

Raja Kumar et al. [41] and Agarwal et al. [42] designed a deep-learning CNN model for predicting tomato leaf diseases in 10 classes. They employed an NVIDIA DGXv100 machine and utilized pre-trained models like VGG16, MobileNet, and InceptionV3. Their dataset included 17,500 images from Plant Village on Kaggle, with a 70/10/20 split for training, testing, and validation. The model achieved an accuracy of 91.2%.

Shrestha et al. [43] developed a CNN model to forecast tomato leaf diseases. They collected 3,000 images from Plant Village on Kaggle, using an 80/20 split for training and testing. Their computer setup included 16 GB DDR4 RAM, an Intel Core i7-9600k processor, and Windows 10. Disease and pest management were not mentioned, and the proposed model achieved an accuracy of 88.80%.

In a study by Sardogan et al. [44], the authors developed a multiclass CNN and LVQ predictive model for tomato leaf diseases using a dataset of 500 images from a plant community on Kaggle. The dataset was split 80/20 for training and testing. The main emphasis of this research is developing a more precise and effective deep learning and machine learning model for the early classification and detection of multi-class tomato leaf diseases.

3. Materials and Methods

This part of the research report thoroughly describes the procedures, equipment, and techniques used to gather and evaluate data to meet the study's goals and respond to its questions. It discusses topics including the methods used to collect data and for analysis.

3.1. Data Collection

The data collection strategy used for this research effort is covered in this section. In order to comprehend the study, the researchers used a dataset downloaded from plant village (kaggle.com). The website Kaggle.com hosts competitions and offers tools for data science and machine learning. It was founded in 2010, and Google later purchased it in 2017.

Along with a community where data scientists and machine learning enthusiasts can cooperate, share their work, and participate in contests, the platform provides datasets for users to practice and learn from. Kaggle has become a well-liked centre for those interested in data-driven initiatives and provides a forum for exchanging information.

3.2. Description of the Dataset

The dataset is 280 MB and comprises ten classes with 1,000 images each, nine classes of diseased tomato leaves and one healthy class of tomato leaves to make a total of 10,000 images that have been sorted into folders for the training set and validation set.

3.3. Data Preprocessing

We scaled and downsized our datasets by reducing their size from 256 to 224 to prevent the errors caused by standard input shapes that neural networks encounter.

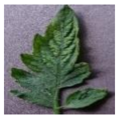
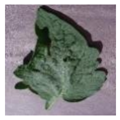
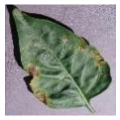
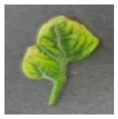
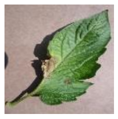
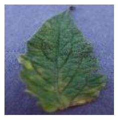
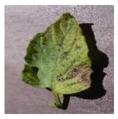
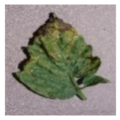
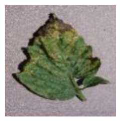
Additionally, some image enhancement was performed, in which each tomato disease image was carefully zoomed at 9%, rotated at a range of 10%, and randomly flipped horizontally because too much zooming is invariant to blur cues in digital images and spatial varying effects where the lines of the edges of digital images look asymmetrical.

Again, augmentation has proven to be the most effective way to improve model performance while avoiding additional expenses. We apply dataset rescaling, shearing, and flipping to capitalise on such a sound statement. We applied a horizontal flip of 50% and a shear range of 20%. Finally, we rescaled the datasets by multiplying by one and dividing by 255 to retain a consistent distribution of pixel values.

3.4. Feature Engineering

MobileNetV2 was used with an input dimension of 224 224 × 3 and a group dimension of 4 to ensure that fine-grained features were extracted from each image in the 10,000 dataset. The feature extraction procedure took about 12 hours to complete.

Table 1. List of tomato leaf disease classes used in this study

S/N	Name of Tomato diseases	Image	Symptoms
1	Mosaic virus		Tiles, wrinkling, leaflet decrease and curvature, and uneven fruit ripening.
2	Tomato target spot		Pinpoint-sized, water-soaked patches on the upper leaf surface, Initial symptoms on stems and petioles are lesions that are somewhat sunken, brown specks.
3	Bacteria spot		Small, spherical, water-soaked dots with a yellow halo that progressively turns dark brown or black. Lesions can range in size, but they rarely grow to be larger than 1/10 inch in diameter.
4	Tomato yellow leaf curl virus		Little leaves with veins that turn yellow. The leaves also curl upward and toward the center of the leaf.
5	Tomato late blight		Tomato leaves are water-soaked, irregularly shaped lesions that frequently have a lighter halo or ring surrounding them.
6	Leaf mold		Initially, the oldest leaves become affected. On the upper sides of leaves, tiny pale greenish-yellow dots with no discernible borders, usually less than 1/4 inch across, develop.
7	Early blight		Older foliage near the ground has tiny black patches. Round, brown, and up to 1/2 inch across, leaf spots are an oblong shape.
8	Tomato red spider mite		The whitening or yellowing of leaves due to tomato red spider mite feeding causes the leaves to dry out and fall off finally.
9	Septoria leaf spot		It is visible on the lower leaves after the first fruit sets. The round spots have dark brown edges, tan to grey cores, and little black fruiting structures. They are between 1/16 and 1/4 of an inch in diameter.
10	Tomato healthy		Healthy

3.5. Train-Test Split

The preprocessed dataset must be divided into training, testing, and evaluation sets. In order to avoid problems with over-fitting brought on by errors and outliers, we also developed a splitting approach where we merge the entire folders containing a total of 10,000 images and use 8,000 (80%) for training the proposed hybrid model (MobileNetv2 + SVM), and 2,000 (20%) for testing the model's performance in order to avoid overfitting.

3.6. Model Selection

We cascaded MobileNetv2 and Support Vector Machine (SVM) algorithms for effective multi-class tomato leaf disease classification and prediction. The proposed cascaded system design is represented in Figure 3, which depicts the authentic design of the internal workings of the suggested model. We employed a transfer learning technique to extract deep enriched features from our training sets of tomato picture samples.

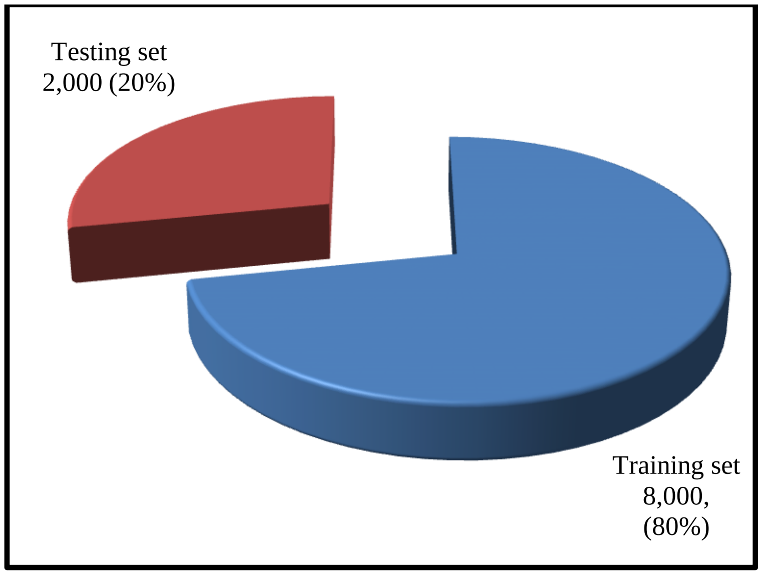


Fig. 2 Dataset splitting ratio

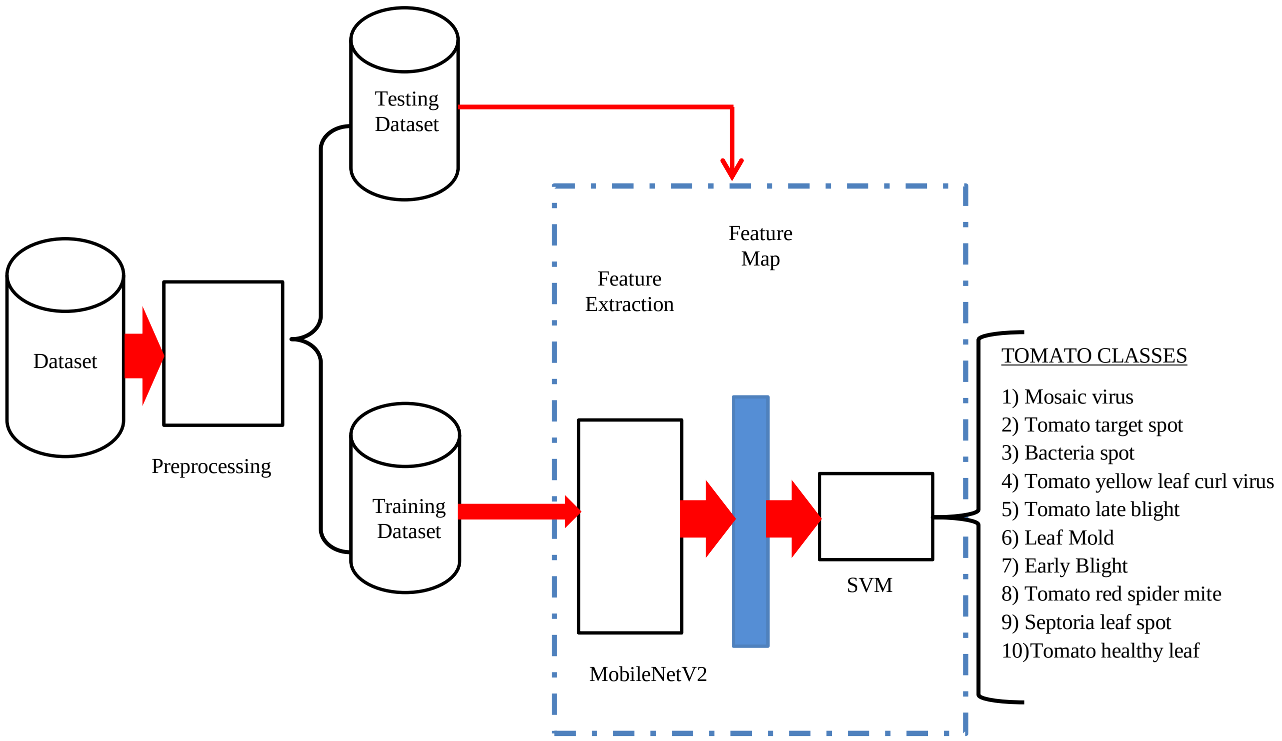


Fig. 3 Proposed system architecture

In order to distinguish the ten classes (including a healthy tomato leaf) clearly, the datasets are examined during the preparation stage for accurate name labels and disease-related masking.

The datasets were divided into different ratios based on different splitting techniques. The primary neural network model was MobileNetv2, a portable State of The Art (SOTA) deep learning model. The input dimension for this model was 224 224 x 3, where 224 stands for the depth and breadth of the training dataset and 3 for the channel (RGB). Using the MobileNet preprocessing function, the training samples were normalized to a scale ranging from 0 to 1.

We solved the feature extraction problem of the network by setting the base_model trainable option to false.

Additionally, the weights are being loaded from a state dictionary that was saved during the training of mobileNet using the about 22k picture ImageNet database.

The size of each extracted feature was 1024 pixels, which equals the overall size of the 2D image. The retrieved features were saved to a Numpy array and then used to train a Support Vector Machine (SVM) to forecast which class the extracted feature belongs to. The SVM is superior to the fully connected layer in classifying multi-class and complex small to medium datasets. This is the reason it was selected as the classification method. SVM was imported using the Sklearn Python Library. We added the hinge loss type to the loss function, which seeks the maximum point between 0 and (t-1). Hinges lose zero and one when t>=1 and when t1, respectively.

Table 2. MobileNetv2 algorithm

```
Algorithm 1: MobileNetv2 Algorithm
Step 1 : Input an RGB picture with the dimensions (224, 224, 3).
Step 2 : Scale the pixel values by a factor of 255 and perform preprocessing on the input image by removing the mean RGB values (R_mean, G_mean, and B_mean).
Step 3 : Apply a series of convolution operations:
    a. A 3x3 kernel, stride 2, and ReLU activation Convolutional Layer (CL).
    b. A 3x3 kernel, stride 1, and ReLU activation are used in depthwise convolution.
    c. 1x1 kernel-based pointwise convolution with ReLU activation.
    d. Repeat steps a, b, and c using various filters and stride settings following the MobileNetV2 design.
Step 4 : Add bottleneck layers (1x1 point-wise convolution followed by 3x3 depth-wise convolution followed by 1x1 point-wise convolution)
Step 5 : Apply batch normalization to increase training stability after CL.
Step 6 : To maintain information flow, use a linear bottleneck at the start of each sequence of layers.
Step 7 : Apply a global average Pooling Layer (PL) to convert the feature map's spatial dimensions to a vector.
Step 8 : Add an FC layer with softmax activation for multi-class categorization in step 8.
Step 9 : Output the class probabilities for each class.
```

Table 3. SVM algorithm

```
Algorithm 2: SVM Algorithm
Step 1 : Input the training dataset consisting of feature vectors and associated class labels.
Step 2 : Select a kernel function (such as a linear, polynomial, or radial basis function) to calculate the similarity between feature vectors.
Step 3 : Preprocess the data by scaling or normalizing the feature vectors.
Step 4 : Construct the SVM optimization problem to identify an ideal hyperplane that optimizes the margin between the various classes.
Step 5 : To identify the support vectors and appropriate hyperplane parameters, solve the optimization problem using quadratic programming or an optimization technique.
Step 6 : Determine the similarity of fresh test samples to the support vectors and classify them according to the learned hyperplane.
Step 7 : To evaluate the effectiveness of the SVM model, use metrics like accuracy, precision, recall, and F1-score.
Step 8 : If necessary, fine-tune the model by changing variables like the regularization or kernel parameters.
Step 9 : Repeat steps 4 through 8 until the performance is satisfactory.
Step 10 : Output the trained SVM model for upcoming forecasts using fresh, unforeseen data.
```

3.7. The SVM Classification Head

There are 1280 different feature shapes in the 8,000 training samples used for the SVM classification, and the reminding 2,000 was used to test the model. It may be challenging to effectively categorize and predict multi-class tomato leaf diseases with ten classes. SVM is designed to deal with binary classifications. Therefore, we have two options for applying the SVM to a multi-class classification task: One vs. All or One vs. One. A one-versus-one multiclass classification method is the classification technique utilized in this research work. The strategy for the one-versus-one approach employed in this study entails creating a classifier to forecast which classes will be positive and which will be negative. Therefore, we used 45 classifiers in all. It was obtained using the formula below.

$$\text{No. of Classifier} = \text{R}(\text{R}-1)\big/_2 \tag{8}$$

Where R is the number of classes involved in the classification task. Hence, substituting our class as 10.

$$\text{No. of Classifier} = 10(10-1)\big/_2 \tag{9}$$

$$\text{No. of Classifier} = 45 \tag{10}$$

The application of 45 smaller Support Vector Machines (SVMs) for binary classification is shown in Figure 4. The class to which an instance, denoted by the symbol X, belongs is determined by combining the predictions derived from various SVMs using majority voting. With this method, you may train effectively without repeating it repeatedly (repetition). The SVC module from the scikit-learn library is imported into Python to implement this strategy. Different splitting techniques and kernel types, such as RBF, Linear, or Gaussian, assure fairness. To evaluate the performance of the SVM, random values are also assigned to the hyperparameter C.

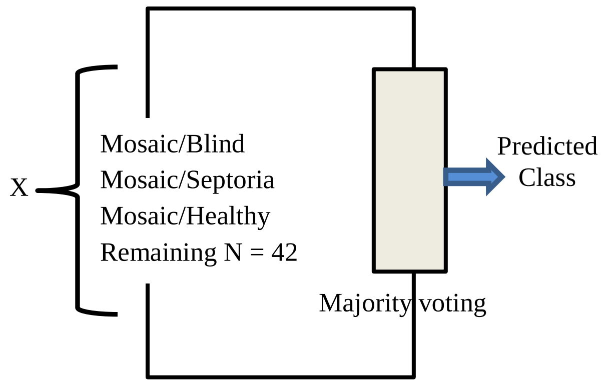


Fig. 4 The SVM classification head diagram

3.8. Model Training

According to the previously recommended splitting techniques, the imported features in the numpy file are read and separated into X_train, Y_train, and X_test, Y_test. We employed the kernel trick to solve the multi-class SVM's overfitting problem.

3.9. Model Evaluation

When assessing the usefulness of the suggested hybrid classification technique, the confusion matrix was found to be a valuable tool. It makes a prediction-to-actual label comparison and outputs the findings as a square matrix. The columns of the matrix correspond to the anticipated labels, whereas the rows of the matrix correspond to the genuine labels. Examining the confusion matrix, we may learn more about how effectively the model identified the various classes. The confusion matrix was created using sklearn, and metrics, and section IV discusses the confusion report that shows how the model classified the 2,000 test dataset. The following metrics were used to evaluate the suggested hybrid model's performance: accuracy, precision, recall, and F1-score.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{11}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{12}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{13}$$

$$\text{F1} - \text{Score} = 2 * \frac{(\text{Recall}*\text{Precision})}{(\text{Recall}+\text{Precision})} \tag{14}$$

3.10. Hyperparameter Tuning

In addition to tuning numerous hyper-parameters, we also considered several kernel variables. In order to assess the model's effectiveness in terms of instance range of influence vs decision boundary, we increased the polynomial kernel's degrees from 3 to 10. The gamma parameters in the 0 to 5 range were also altered using the Gaussian RBF kernel. The design of our prototype was based on the mixture that produced the best results. The recall and precision values and accuracy graphs were all recorded. The ROC graph displayed the results as we looked at the tradeoffs between precision and recall.

3.11. Experimental Platform

The research used a test setup with an 8GB GDDR5X NVIDIA GeForce GTX 1080Ti GPU, 64GB DDR4 2133MHz RAM, and an Intel Core i7-8700 CPU at 3.2–4.6GHz. The platform employed TensorFlow 10.0 as the framework, Python 3.5 as the development language, and Linux Ubuntu 21.04 as the operating system, as shown in Table 4. This setup offered a robust and reliable environment for conducting experiments and successfully training the model.

4. Results and Discussion

The results of the investigation are provided and thoroughly examined in this section. The gathered data is evaluated, and the consequences of the findings are examined. They are contrasted with prior research and theoretical predictions to comprehend the results' relevance fully. The confusion matrix provides information on how many predictions were correct and what percentages of predicted classes were correct. It also shows how frequently a prediction is correct. The generator classes named Y_test_label were set to be the original classes of the test samples. The Y_pred_label shows the output of the prediction.

Table 5 displays the proposed model summary metrics for each more miniature Support Vector Machine (SVM) used for binary classification. There are 45 classifiers in total; however, only the findings of 7 binary classifiers are shown in order to enhance reading and comprehension by omitting extraneous information. Results involving back iteration and comparing the negative classes are not displayed.

The ROC curve was used to show the connection between the true and false positive rates. The true positive rate is contrasted using the ROC curve (also known as recall) to the false positive rate. The FPR calculates the percentage of adverse occurrences mistakenly labelled as favourable. The genuine negative rate, which is one minus that quantity, is the proportion of cases that are accurately classified as negative. Specificity is another name for the True Negative Rate (TNR). The ROC curve thus depicts the link between sensitivity (recall) and 1-specificity. The graphical diagram in Figure 6 was produced by passing FPR and TPR to the roc_curve function and plotting the outcome with Matplotlib.

The ROC curve in Figure 6 also illustrates the tradeoff between recall and precision. While a randomly chosen classifier will have a ROC AUC value of 0.5, a successful classifier will have one close to or equal to 1.

Table 4. Configuration of experimental platform parameters

Configuration	Parameters
CPU	Intel® Core ™ i7-8700, CPU@3.2 – 4.6GHz
GPU	NVIDIA GeForce GTX 1080Ti 8GB GDDR5X
Memory (RAM)	64GB DDR4 2133MHz
Operating System (OS)	Linux Ubuntu 21.04
Development Language	Python 3.5
Development Platform	Anaconda3, Tensorflow 10.0

Table 5. SVM classification heads metric results

SVMs Head	Precision	Recall	Accuracy	F1-Score
Mosaic/Blindspot	99%	100%	99%	100%
Blindspot/Bacteria spot	92%	92%	91%	92%
Bacteria spot/Septoria	87%	91%	92%	89%
Septoria/Yellow_Leaf	94%	89%	89%	92%
Yellow_Leaf/Late_Blight	85%	89%	88%	88%
Late_Blight/Leaf_Mold	94%	86%	93%	90%
Leaf_Mold/Early_Blight	99%	100%	99%	99%

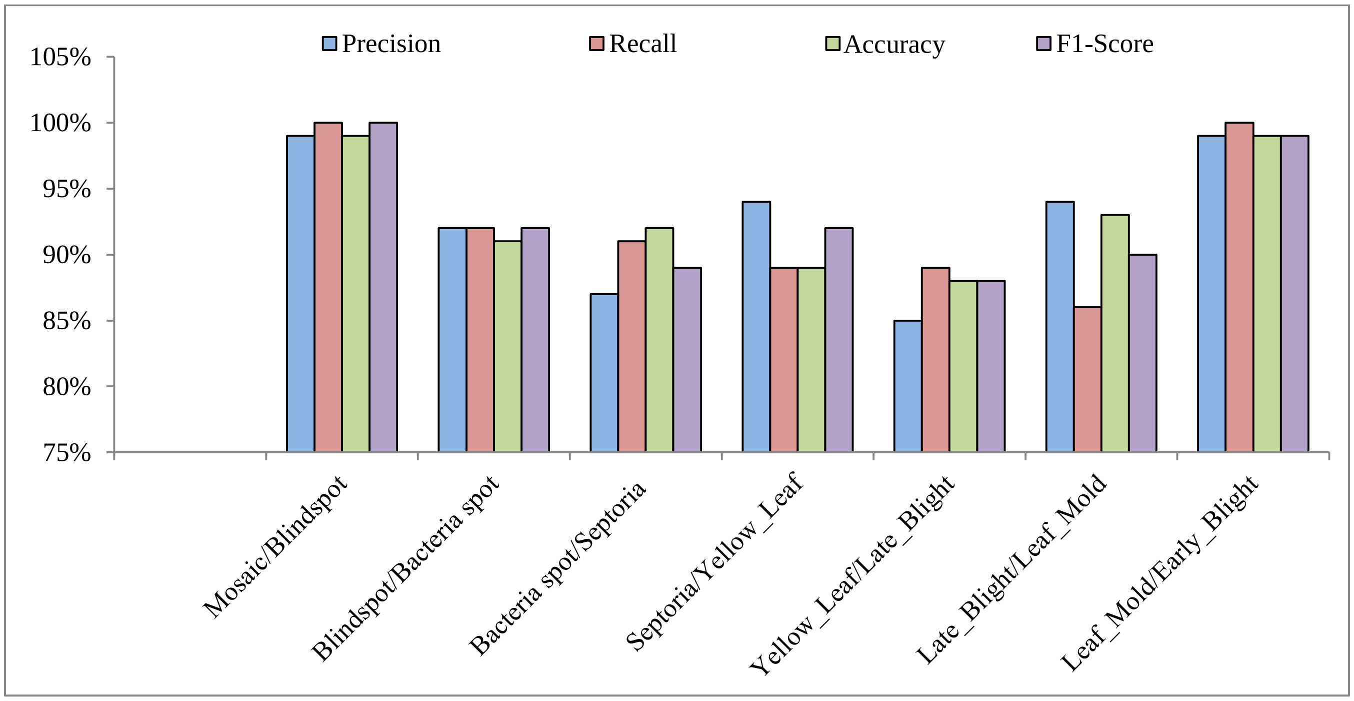


Fig. 5 Column diagram depicting SVM classification head result

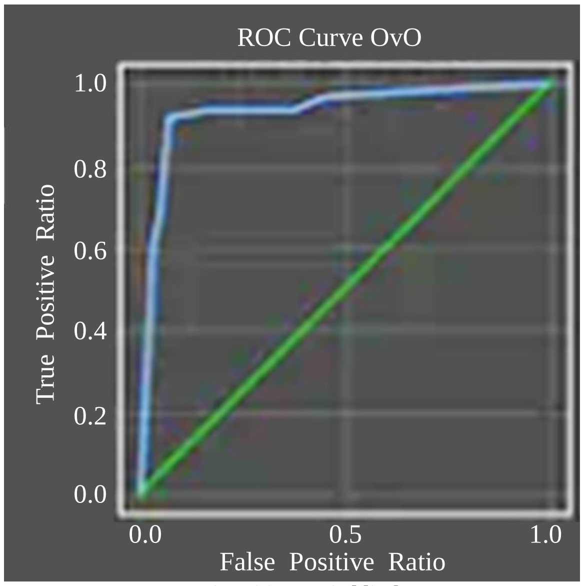


Fig. 6(a) Mosaic/blind spot

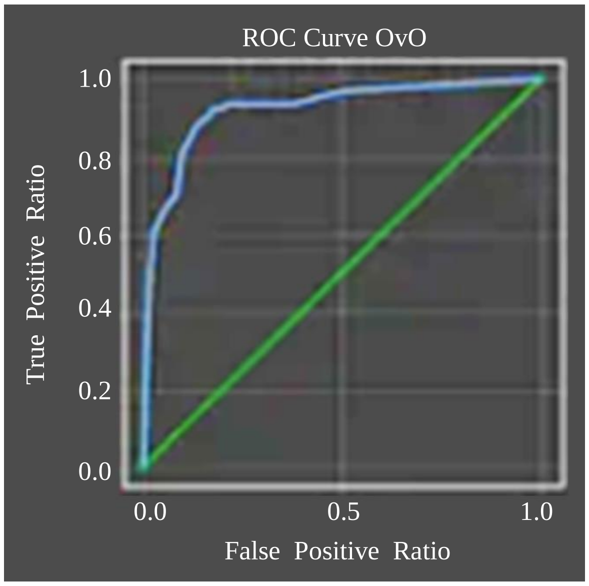


Fig. 6(b) Late_blight/leaf mold

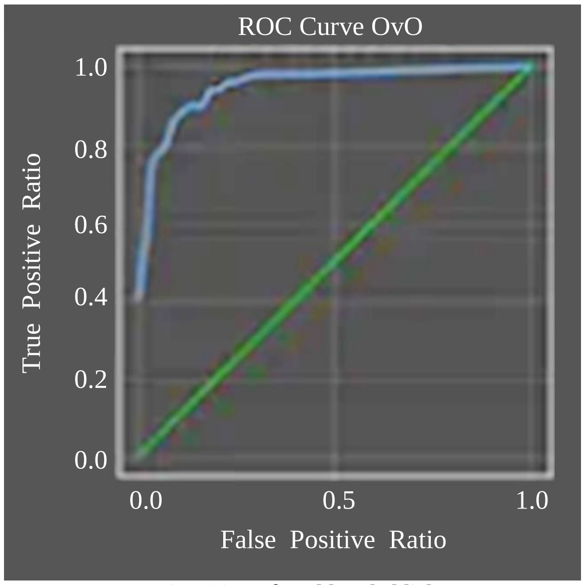


Fig. 6(c) Leaf_mold/early blight

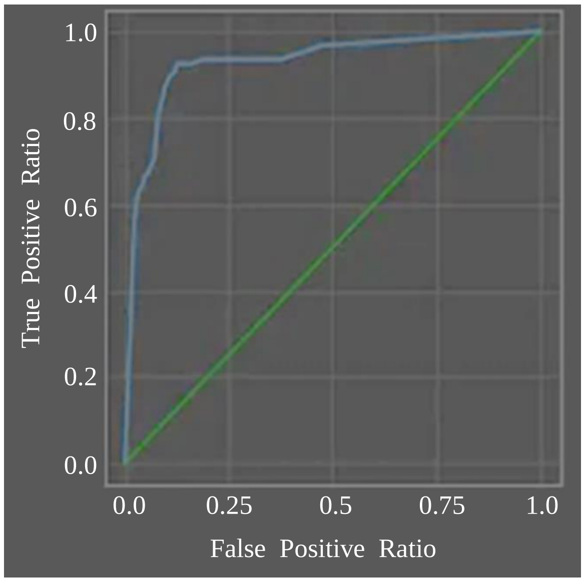


Fig. 7 The average 45 classifiers' ROC curve

Figures 6(a), (b), and (c) are the ROC Curve for three identified classifiers (small SVMs). They also represent some identified ROC values with accuracy equal to 99%, 93% and 99%, respectively. While Figure 7 shows the average ROC curve for the entire SVM as a whole.

The combined full mini or smaller classifiers' average accuracy is given 95% average accuracy. The generated average accuracy ROC shown in Figure 7 illustrates the relationship and behavior between the genuine and false positive classes. The proposed model, which combines MobileNetv2 with SVM, achieved 99% accuracy, as shown in Table 6 and Figure 8.

Table 6. Result of the model's accuracy comparison

Model	Accuracy
Proposed Model (MobileNetv2 + SVM)	99%
CNN (Filter size: 3x3)	95%
CNN (Filter size: 5x5)	94%

In contrast, the experiment using the CNN model based on (3x3) and (5x5) filter sizes had 95% and 94% accuracy, respectively. These results show the improved classification and identification accuracy of the proposed model.

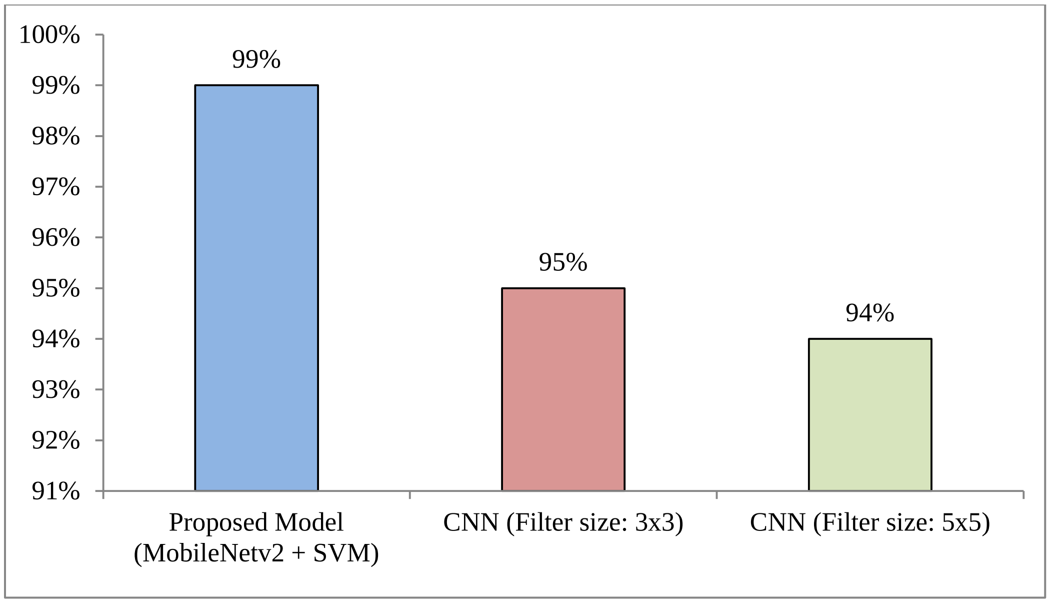


Fig. 8 Models accuracy column diagram

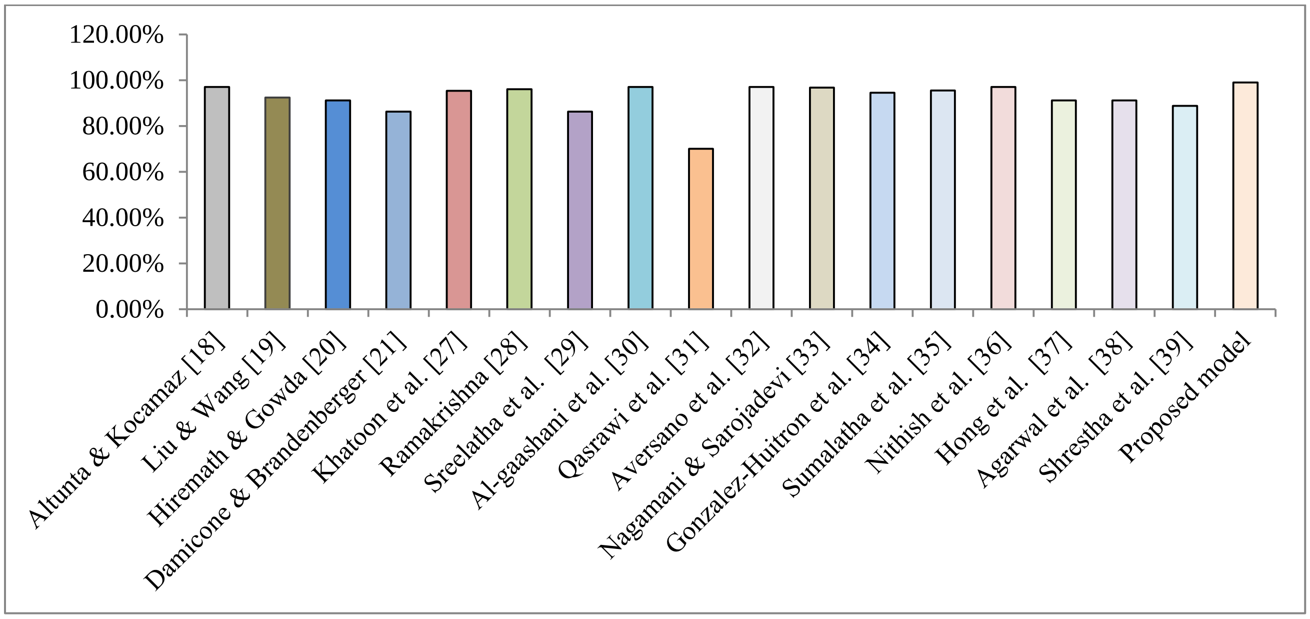


Fig. 9 Accuracy comparison column diagram

Figure 9 displayed a column diagram showing the accuracy comparison with a few other models used in this study. The accuracy range for the proposed model, which ranged from 88% to 99%, surpassed that of other relevant models ranging from 70% to 97%. Findings demonstrate the suggested model's superior performance over the other models in correctly classifying the images, demonstrating its fit for the task at hand.

5. Conclusion

This study proposes a hybrid model employing deep learning methods for categorizing and predicting tomato leaf diseases. The suggested model combines Support Vector Machine (SVM) methods with MobileNetv2, a deep learning architecture. Ten thousand tomato leaf photos were used in the study dataset, divided into ten classes for damaged leaves and one for healthy leaves. Data is gathered from Kaggle, scaled and enhanced using data preprocessing, and fine-grained features are extracted using feature engineering using MobileNetv2.

The hybrid model is trained using 80% of the dataset, and its effectiveness is assessed using the remaining 20%. The hybrid model architecture is depicted in Figure 3, along with the integration of the MobileNetv2 and SVM algorithms. MobileNetv2 is the primary neural network model utilized for feature extraction, and SVM is employed for multi-class classification. Utilizing MobileNetv2's pre-trained weights on the ImageNet database requires transfer learning.

The study concludes that a helpful strategy for the early classification and detection of tomato leaf diseases is the hybrid model that combines MobileNetv2 and SVM. The model shows promising performance in accurately diagnosing and forecasting numerous diseases affecting tomato plants by utilizing deep learning and machine learning approaches. The model is more accurate and resilient based on incorporating SVM and using pre-trained weights from MobileNetv2.

The research emphasizes the significance of utilizing cutting-edge technology for agricultural applications, such as deep learning and machine learning. By utilizing a hybrid model that combines deep learning for feature extraction and machine learning for classification, the research provides farmers with a valuable tool to accurately diagnose and cure tomato leaf illnesses. The results of this study contribute to the field of agricultural technology and provide workable strategies for increasing crop output and maintaining food security. The study also suggests several crucial future research directions. Firstly, expanding the dataset by collecting more diverse tomato leaf images can improve the model's ability to classify various diseases accurately. Secondly, exploring alternative deep learning architectures and novel models designed for agricultural applications can enhance classification performance. Thirdly, correcting class imbalance through data augmentation or undersampling strategies might increase the model's accuracy for underrepresented classes. Fourthly, incorporating temporal information through video or time-series analysis can capture disease progression and enable early detection.

Extending the proposed methodology to other crops and deploying it in real-world scenarios, such as on-field applications, can validate its performance and practicality. Finally, integrating the hybrid model with precision agriculture technologies can enable real-time monitoring and targeted interventions, improving disease management and optimising crop productivity. These research directions can advance agricultural technology, enhance disease management practices, and support sustainable agriculture.

References

[1] Tiago Domingues, Tomás Brandão, and João C. Ferreira, "Machine Learning for Detection and Prediction of Crop Diseases and Pests: A Comprehensive Survey," *Agriculture*, vol. 12, no. 9, pp. 1-23, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[2] Dániel Fróna, János Szenderák, and Mónika Harangi-Rákos, "The Challenge of Feeding the World," *Sustainability*, vol. 11, no. 20, pp. 1-18, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Iftikhar Ahmad et al., "Optimizing Pretrained Convolutional Neural Networks for Tomato Leaf Disease Detection," *Complexity*, vol. 2020, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[4] Max Roser et al., "World Population Growth," *Our World in Data*, 2013. [Online]. Available: http://www.sisterschiropractor.com/world-population-growth.html#citation

[5] Wimpie Nell et al., "A Creative Multidisciplinary Approach Towards the Development of Food Gardening," *Development Southern Africa*, vol. 17, no. 5, pp. 807-819, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[6] Deepak K. Ray et al., "Yield Trends are Insufficient to Double Global Crop Production by 2050," *PLoS One*, vol. 8, no. 6, p. 0066428, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[7] Mary Izuaka, Traders Report a Tomato Shortage as a Pest Reappears in Kano, Premium Times, 2020. [Online]. Available: https://www.premiumtimesng.com/agriculture/agric-news/394536-traders-report-tomato-shortage-as-pest-reappears-in-kano.html

[8] Food and Agriculture Organization of the United Nations, The Future of Food and Agriculture: Trends and Challenges, Rome, 2017. [Online]. Available: https://www.fao.org/3/i6583e/i6583e.pdf

[9] C. P. Adekunle, C. I. Alarima, and E. T. Tolorunju, "Differentials in Willingness to Pay for Biofortified Tomato Fruits in Abeokuta Metropolis, Ogun State, Nigeria," *Nigeria Agricultural Journal*, vol. 50, no. 2, pp. 1-9, 2019. [Google Scholar] [Publisher Link]

[10] Minse Modi, Albert Modi, and Sheryl Hendriks, "Potential Role for Wild Vegetables in Household Food Security," *African Journal of Food, Agriculture, Nutrition, and Development*, vol. 6, no. 1, pp. 1–13, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[11] Debjit bhowmik et al., "Tomato: A Natural Medicine and Its Health Benefits," *Journal of Pharmacognosy and Phytochemistry*, vol. 1, no. 1, pp. 33-43, 2012. [Google Scholar] [Publisher Link]

[12] Wilfried Baudoin et al., Good Agricultural Practices for Greenhouse Vegetable Production in the Southeast European Countries, 2017. [Online]. Available: http://www.fao.org/3/a-i6787e.pdf

[13] Food and Agriculture Organization of the United Nations, Integrated Pest Management in Tomatoes in Eritrea In Sustainable Management of Arthropod Pests of Tomato, Rome, 2019. [Online]. Available: https://www.fao.org/3/ca6480en/CA6480EN.pdf

[14] Illaria Buja et al., "Advances in Plant Disease Detection and Monitoring: From Traditional Assays to In-Field Diagnostic," *Sensors*, vol. 21, no. 6, pp. 1-22, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[15] A. M. John-Otumu, G. U. Ogba, and O. C. Nwokonkwo, "A Survey on Artificial Intelligence based Techniques for Diagnosis of Hepatitis Variants," *Journal of Advances in Science and Engineering*, vol. 3, pp. 43-56, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[16] A. M. John-Otumu et al., "AI-based Techniques for Online Social Media Network Sentiment Analysis: A Methodical Review," *International Journal of Computer and Information Engineering*, vol. 16, no. 12, pp. 550-560, 2022. [Google Scholar] [Publisher Link]

[17] Boran Sekeroglu, and Ilker Ozsahin, "Detection of Covid-19 from Chest X-Ray Images using Convolutional Neural Networks," *SLAS Technology: Translating Life Sciences Innovation*, vol. 25, no. 6, pp. 553-565, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[18] Y. Altunta, and F. Kocamaz, "Deep Feature Extraction for the Detection of Tomato Plant Diseases and Pests Based on Leaf Images," *Celal Bayar Üniversitesi Fen Bilimleri Dergisi*, vol. 17, no. 2, pp. 145-152, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[19] Jun Liu Xuewei Wang, "Tomato Diseases and Pest Detection Based on an Improved Yolo V3 Convolutional Neural Network," *Frontiers in Plant Science*, vol. 11, pp. 1-12, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[20] Vinutha B. Hiremath, and M. M. Sandarsh Gowda, "Disease Prediction of Tomato Leaf using CNN and Deep Learning Techniques," *International Research Journal of Modernization in Engineering, Technology, and Science*, 2020. [Google Scholar]

[21] John P. Damicone, and Lynn Brandenberger, "Common Diseases of Tomatoes: Part I. Diseases Caused by Fungi," *Oklahoma Cooperative Extension Service*, pp. 1-6, 2016. [Publisher Link]

[22] Abrham Debasu Mengistu, Seffi Gebeyehu Mengistu, and Dagnachew Melesew, "An Automatic Coffee Plant Diseases Identification using Hybrid Approaches of Image Processing and Decision Tree," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 9, no. 3, pp. 806-811, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[23] M. P. Islam et al., "Performance Prediction of Tomato Leaf Disease by a Series of Parallel Convolutional Neural Networks," *Smart Agricultural Technology*, vol. 2, p. 100054, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[24] Lilian Mkonyi et al., "Early Identification of Tuta Absoluta in Tomato Plants using Deep Learning," *Scientific African*, vol. 10, p. e00590, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[25] M. M. Gunarathna, R. M. K. T. Rathnayaka, and W. M. W. Kandegama, "Identification of an Efficient Deep Leaning Architecture for Tomato Disease Classification using Leaf Images," *Journal of Food and Agriculture*, vol. 13, no. 1, pp. 33-53, 2020. [Google Scholar] [Publisher Link]

[26] Naresh K. Trivedi et al., "Early Detection and Classification of Tomato Leaf Disease using a High-Performance Deep Neural Network," *Sensors*, vol. 21, no. 23, pp. 1-15, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[27] Shaheen Khatoon et al., "Image-based Automatic Diagnostic System for Tomato Plants using Deep Learning," *Computers, Materials, and Continua*, vol. 67, no. 1, pp. 595-612, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[28] Rajath Ramakrishna, "A Machine Learning-Based Approach to the Detection and Classification of Tomato Plant Leaf Diseases," *NORMA eResearch @NCI Library*, 2020. [Publisher Link]

[29] P. Sreelatha et al., "Managing Tomato Leaf Disease Detection Accuracy using Computer Vision-Based Deep Neural Network," *Journal of Contemporary Issues in Business and Government*, vol. 27, no. 1, pp. 3425–3437, 2021. [Google Scholar] [Publisher Link]

[30] Mehdhar S. A. M. Al-gaashani et al., "Tomato Leaf Disease Classification by Exploiting Transfer Learning and Feature Concatenation," *IET Image Processing*, vol. 16, no. 3, pp. 913-925, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[31] Radwan Qasrawi et al., "Machine Learning Techniques for Tomato Plant Disease Clustering, Prediction, and Classification," *In Proceedings of the 2021 International Conference on Promising Electronic Technologies*, Palestine, pp. 40–45, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[32] Lerina Aversano et al., "Tomato Disease Classification based on VGG and Transfer Learning," *2020 IEEE International Workshop on Metrology for Agriculture and Forestry*, Italy, pp. 129-133, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[33] H. S. Nagamani, and H. Sarojadevi, "Tomato Leaf Disease Detection using Deep Learning Techniques," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 1, pp. 305-311, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[34] D. Sheema et al., "The Detection and Identification of Pest-FAW Infestation in Maize Crops using Iot-Based Deep-Learning Algorithm," *SSRG International Journal of Electrical and Electronics Engineering*, vol. 9, no. 12, pp. 180-188, 2022. [CrossRef] [Publisher Link]

[35] Victor Gonzalez-Huitron et al., "Disease Detection in Tomato Leaves via CNN with Lightweight Architectures Implemented in the Raspberry Pi4," *Computers and Electronics in Agriculture*, vol. 181, p. 105951, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[36] G. Sumalatha, Jhansi Rani Singothu, and S. Krishna Rao, "Transfer Learning-based Plant Disease Detection," *International Journal for Innovative Engineering and Management Research*, vol. 10, no. 3, pp. 469-477, 2021. [Google Scholar] [Publisher Link]

[37] Nithish Kannan E et al., "Tomato Leaf Disease Detection using A Convolutional Neural Network with Data Augmentation," *Proceedings of the 5th International Conference on Communication and Electronics Systems*, pp. 1125-1132, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[38] Vatsal Mahajan, Dilip Jain, and Abhinav Dua, "Plant Leaf Segmentation Invariant of Background," *International Journal of Computer & Organization Trends*, vol. 4, no. 5, pp. 15-18, 2014. [CrossRef][Publisher Link]

[39] Huiqun Hong, Jinfa Lin, and Fenghua Huang, "Tomato Disease Detection and Classification by Deep Learning," *Proceedings of the 2020 International Conference on Big Data, Artificial Intelligence, and Internet of Things Engineering*, China, pp. 25-29, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[40] Gargi Sharma, and Gourav Shrivastava, “Crop Disease Prediction using Deep Learning Techniques - A Review,” *SSRG International Journal of Computer Science and Engineering*, vol. 9, no. 4, pp. 23-28, 2022. [CrossRef] [Publisher Link]

[41] R. Raja Kumar et al., “Novel Segmentation and Classification Algorithm for Detection of Tomato Leaf Disease,” *Concurrency and Computation: Practice and Experience*, vol. 335, no. 12, p. e7674, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[42] Mohit Agarwal et al., “ToLeD: Tomato Leaf Disease Detection using Convolutional Neural Network,” *Procedia Computer Science*, vol. 167, pp. 293-301, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[43] Garima Shrestha et al., “Plant Disease Detection using CNN,” *Proceedings of the 2020 IEEE Applied Signal Processing Conference*, pp. 109-113, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[44] M. Sardogan, A. Tuncer, and Y. Ozen, “Plant Leaf Disease Detection and Classification based on CNN with the LVQ Algorithm,” *UBMK 2018: 3rd International Conference on Computer Science and Engineering*, pp. 382-385, 2018. [CrossRef] [Google Scholar] [Publisher Link]