A Deep Learning Approach in Tomato Leaf Disease Identification and Classification using DenseNet-121 model

Babul Sarkar[1] and Kuldip Kaityar[2]

Department of Mathematics, Chandigarh University, Mohali, Punjab 140413, India,
[1]babulsarkaar633@gmail.com,[2]kuldeepe7279@cumail.in

Abstract. In this paper, we will see how good a convolutional neural network (CNN) works in identifying various diseases in a tomato plant and then classifying them accordingly by analyzing the images of its leaves. The specific deep learning model used here for this approach is DenseNet-121 (i.e., Dense Convolutional Neural Network-121), which is very efficient in image classification as it uses 121 layers of modification for each input, i.e., image, in order to train itself. Two open-source datasets, 'PlantVillage' and 'TomatoLeaf', are combined together to be used as the final dataset in training and validating the model. The overall classification precision achieved by the proposed model reaches an approximate value of 99.71%, which shows how effective and efficient the DenseNet-121 model can be in disease detection and classification for a tomato plant. This also shows that traditional methods like manual inspections should upgrade themselves in adapting to such deep learning methods in saving time and energy for disease detection in tomato plants with much better accuracy.

Keywords: Tomato Production, Diseases, Deep Learning, DenseNet-121, CNN

1 Introduction

Today, agriculture in India is responsible for engaging around 45% of the country's workspace and eventually contributing around 16% to the country's GDP and 10% of total exports [1]. In addition to that, production of tomatoes in India alone makes up to 10% of the entire tomato production in the world [2], making the country the second largest tomato producer in the world (after China) [3]. One of the southernmost states of India, Tamil Nadu is among the top tomato growing states in the country, producing around 720,000 metric tonnes of tomatoes from around 44,000 hectares of crop area [2]. With massive production and an increase day by day in tomato production, the risk of diseases in crops also increases. The major diseases in tomato crop with the approximate amount of crop loss caused by them in Tamil Nadu alone are early blight (50% - 86%), late blight (20% - 70%), tomato fruit borer (85% - 93.7%), root knot nematode (27%), yellow-leaf-curl virus (35%), mosaic virus (18%), bacterial leaf spot (20%

- 50%) and many more [2] [4] [5]. Furthermore, around 27% tomato crops in New Delhi are damaged due to excessive use (25-30 spray rounds) of chemical cocktail pesticides in a season, leading to various adverse effects mentioned above, according to the baseline survey conducted by the National Research Center for Integrated Pest Management (NCIPM) of the Indian Council of Agriculture Research (ICAR) [5].

Tomatoes *(Solanum lycopersicum L.)* are among the most significant and cultivated vegetable crops around the world and are also used as model plants for genetic research on fruit quality, tolerance to biotic and abiotic stress, and other physiological characteristics [6] [7]. Therefore, prevention of diseases and loss in the production of these crops is more than important for a sustainable agricultural environment as well as human development.

2 Literature Review

Authors used various Deep Learning models for disease identification and classification. For instance, [8] used AlexNet and GoogleNet CNN model in identification of soybean plant diseases, while [9] introduced a custom-made CNN model based on deep learning techniques like convolutional layers, Inception-like modules, and dense layers to detect diseases in different plant leaves including tomato plants.

With architectures like ResNet, VGG, Inception, and custom CNNs being used, recent developments in deep learning have demonstrated significant success in the detection of plant diseases. Despite the high accuracy of these models, DenseNet-121 has unique benefits that make it the best choice for this project:

- Dense connectivity allows to reuse the feature layers, improves gradient flow and enables the model to learn more robust features with fewer parameters.
- DenseNet-121 has a favorable balance of accuracy and computational cost compared to deeper networks like ResNet-101 or Inception-v3.
- Previous works achieved strong results with DenseNet-121 in crop and leaf disease classification tasks, reinforcing its reliability in this domain.

Despite the success of alternative models [9] [10], proposed DenseNet-121 model demonstrated its potential for fine-grained leaf disease classification by surpassing these baselines with a validation accuracy of 99.71%.

3 Methodology

The technique used here is a deep learning technique that falls under machine learning, which automatically recognizes and models intricate patterns in data using multi-layered neural networks, or deep neural networks. Deep learning, which draws inspiration from the layered processing structure of the human brain, allows models to directly extract and represent pertinent characteristics

from raw data, in contrast to standard approaches that call for constructed features. Each layer of these networks can capture increasingly complicated representations because artificial "neurons" arranged in interconnected layers convey data through hierarchical structures. Layered architectures of deep learning networks capture complex data relationships to achieve remarkable accuracy and insights across diverse applications like speech recognition, natural language processing, autonomous systems, etc. and for our interest, image recognition, i.e., identification of tomato plant diseases [10] [11].

3.1 Convolutional Neural Network

This is a deep learning architecture that performs very well for processing grid-like data, like images. CNNs capture spatial elements like edges, textures, and forms at various levels by automatically learning hierarchical patterns in data through layers of convolutional filters. They are therefore quite successful at tasks like segmentation, object detection, and image classification [12].

Key aspects of CNNs include [3]:

- Convolutional Layers - These help in applying filters to input data for detecting local patterns and reducing the need for extensive manual feature extraction.
- Pooling Layers - These reduce spatial dimensions, which lowers computation and helps make the network invariant to small translations.
- Fully Connected Layers - Usually towards the end, these layers combine features to make final predictions.

The CNN model used here is DenseNet-121 and it is trained using Jupyter Notebook and Keras API of Tensorflow [13].

3.2 Selection of Appropriate Dataset

For the detection and identification of the diseases a combination of two open-source datasets is used: 'PlantVillage + TomatoLeaf' dataset. This combination of two well-processed datasets helps in training and validating the model to give good results.

This dataset contains 11 classes: 10 classes labeled with the particular types of diseases in tomato plants, e.g., Powdery Mildew, Early Blight, Mosaic Virus, etc. and 1 class labeled as healthy class. Each of these classes are contained in Training and Validation folders with different number of files, i.e. images, belonging to each folder, i.e. each class. Consider Figure 1, showing one out of many images from each of the 11 classes in the training dataset.

The dataset includes a total of **22,878 images**, containing 18,278 *(≈79.89%)* training images & 4,600 *(≈20.10%)* validation images, which were used in the training and validation processes. Table 1 and Table 2 show the distribution of dataset images among the 11 classes for Training and Validation, separately.

(a) Powdery Mildew

(b) Yellow Leaf Curl Virus

(c) Mosaic Virus

(d) Target Spot

(e) Septoria Leaf Spot

(f) 2-spotted Spider Mite

(g) Late Blight

(h) Leaf Mold

(i) Early Blight

(j) Bacterial Spot

(k) Healthy

Fig. 1: Sample images representing each class/category in the training dataset.

Table 1: Training Dataset

Class	Disease	No. of Images
1.	Powdery Mildew	1004
2.	Yellow Leaf Curl Virus	4486
3.	Mosaic Virus	1032
4.	Target Spot	1330
5.	Septoria Leaf Spot	1621
6.	Two-spotted Spider Mite	1545
7.	Late Blight	1740
8.	Leaf Mold	1079
9.	Early Blight	1075
10.	Bacterial Spot	1892
11.	Healthy	1474
Total		18278

Table 2: Validation Dataset

Class	Disease	No. of Images
1.	Powdery Mildew	252
2.	Yellow Leaf Curl Virus	975
3.	Mosaic Virus	218
4.	Target Spot	339
5.	Septoria Leaf Spot	407
6.	Two-spotted Spider Mite	372
7.	Late Blight	502
8.	Leaf Mold	293
9.	Early Blight	298
10.	Bacterial Spot	593
11.	Healthy	351
Total		4600

3.3 Pre-Processing the Data

Before utilizing data for analysis or modeling, it is essential to ensure that the data is clean, structured, and formatted appropriately. This process of preparing raw data to meet the specific requirements of our deep learning model is known as pre-processing of data, which is an important step in any deep learning technique.

In this case, the two datasets used, 'PlantVillage' and 'TomatoLeaf', contain high-quality, carefully selected images which have already undergone thorough pre-processing.

Here, pre-processing refer to the fact that the DenseNet-121 model have already been pre-trained under the ImageNet dataset having 1.2 million images across 1000 categories. All pixel values of the images are re-scaled by a factor of 1/255 to normalize the image data. This pre-processing step of re-scaling the images transforms the pixel values from a range of [0, 255] to a much lesser range of [0, 1], which helps in faster and more stable model training by standardizing the input data. These datasets are therefore free from issues like noise or inconsistencies. This allows the focus to shift directly to model training and evaluation, highlighting the value of using standardized, ready-to-use datasets for efficient and accurate analysis.

3.4 Data Augmentation

Data augmentation means modifying and changing the orientation of input data before exposing it to the model in order to artificially expand the size and diversity of a training dataset. This improves the model's training and eventually increases the chances of getting better results during testing the model on new and unseen datasets, and greatly improving model's performance.

Here, the Sequential class from Keras API of Tensorflow is used for data augmentation of the images of training dataset. Following are the augmentation techniques used after re-scaling the image data:

– **Rotation** *(RandomRotation(0.2))*
This parameter randomly rotates images by up to 20 degrees in either direction. It helps the model become invariant to slight rotations in the input data, allowing it to recognize tomato leaf patterns even if the leaves are slightly tilted. Consider Figure 2.

Fig. 2: Image before & after random rotaion.

– **Translation** *(RandomTranslation(0.1, 0.1))*
This randomly shifts the image vertically and horizontally by up to 10% of its height and width, respectively. This augmentation simulates variations in the leaf's position within the frame, helping in training the model to recognize disease features even when the leaf's vertical and horizontal positions vary in the image, and ultimately helping the model to generalize better to images where the leaf may not be perfectly centered. Consider Figure 3.

Fig. 3: Image before & after random translation

– **Zoom** *(RandomZoom(0.2))*
The zoom augmentation randomly zooms in or out on the image by up to 20%. This simulates variations in the distance from which the leaf image was captured, allowing the model to adapt to different zoom levels and scales. Consider Figure 4.
– **Random Flip** *(RandomFlip("horizontal_vertical"))*
This parameter randomly flips the image horizontally and vertically. The model learns to recognize features irrespective of their orientation, which is

useful as leaf images can be captured from various perspectives. Consider Figure 5.

Fig. 4: Image before & after random zoom.

Fig. 5: Image before & after random flipping

Applying these augmentation techniques increases the diversity of training dataset by generating new and modified versions of the original images, aiming to create a more effective and adaptable model for the required disease detection tasks. This process helps in:

- Reducing Over-fitting: By exposing the model to a variety of transformations, it learns to generalize better and is less likely to memorize specific features of the training set.
- Improving Robustness: The model becomes more robust, i.e., develops greater adaptability, to variations in image capture conditions, like differences in camera-angle, lighting, positioning of leaves, etc.
- Enhancing Generalization: It allows the model to learn from a wider range of scenarios, improving its performance on unseen test data.

3.5 Model Downloading

The DenseNet model variant used here involves downloading all the 121 layers from a pre-trained network (consider [2]) and we benefit from the knowledge

learned on a large dataset 'ImageNet', and adapt it to our specific task with a total number of parameters of 7,037,504.

Table 3 outlines the proposed model design/architecture, combining a pre-trained DenseNet-121 as a frozen feature extractor with a custom classification head for 11 classes.

Table 3: Proposed DenseNet-121 Architecture

Layer Type	Output Size	Details
Input	$256 \times 256 \times 3$	RGB image input
Convolutional Base	$1 \times 1 \times 1024$	Pre-trained on ImageNet + Global Avg. Pooling (trainable=False)
BatchNormalization	$1 \times 1 \times 1024$	Normalizes features from the convolutional base
Dense	$1 \times 1 \times 256$	A fully connected layer utilizing ReLU activation function
Dropout	$1 \times 1 \times 256$	50% dropout for regularization
BatchNormalization	$1 \times 1 \times 256$	Normalizes features from the previous dense layer
Dense	$1 \times 1 \times 256$	A fully connected layer utilizing ReLU activation function
Dense (Output)	$1 \times 1 \times 11$	A fully connected layer utilizing softmax activation function

DenseNet-121, pre-trained on ImageNet, processes input images and outputs feature maps via global average pooling. A custom head with two dense layers (256 units, ReLU activation) and batch normalization is added for classification, incorporating dropout for regularization. The final layer applies softmax activation to output class probabilities.

This approach efficiently leverages pre-trained features while fine-tuning only a few parameters for the specific task.

4 Result and Analysis

After completing 50 epochs of training and validation on the combined 'PlantVillage' and 'TomatoLeaf' dataset, the model demonstrated exceptional performance. The highest training accuracy achieved was 0.9894, with a minimal training loss of 0.0325. Similarly, the highest validation accuracy was 0.9957, with an impressively low validation loss of 0.0149. These metrics indicate that

the DenseNet-121 model has successfully learned from the training data while working effectively on unseen validation data.

For understanding the model's performance visually over the epochs, refer to Figure 6, which illustrates the relationship between training and validation accuracy over the 50 epochs. Correspondingly, Figure 7 provides a plot depicting the changes in training and validation loss over the same period, highlighting model's steady convergence towards an optimal state.

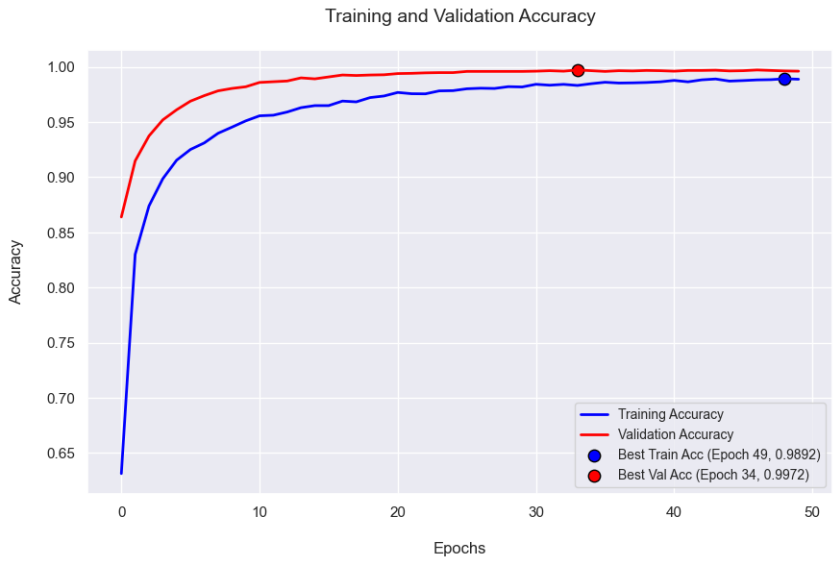

Fig. 6: Training & Validation Accuracy v/s Epochs plot

To further evaluate the performance of the model, a confusion matrix is generated after the completion of training & validation processes, shown in Figure 8. This matrix provides detailed results of the classification performance across different categories by the model. 'True Labels' represent the actual class/category of any image which is provided to the model for testing and 'Predicted Labels' represent the class/category of that image predicted by the model. In simple words, the closer the confusion matrix is to a diagonal matrix, the more accurate the model is in its predictions.

Additionally, a classification report, summarizing the values of recall, precision, F1-score, and support for each class, is presented in Figure 9, providing a thorough overview of the overall performance and effectiveness of the model.

After getting the accuracy results, let us compare our proposed DenseNet-121 model with the previously used deep learning models and their datasets for disease detection in various plant species (consider Table 4):

Fig. 7: Training & Validation Loss v/s Epochs plot

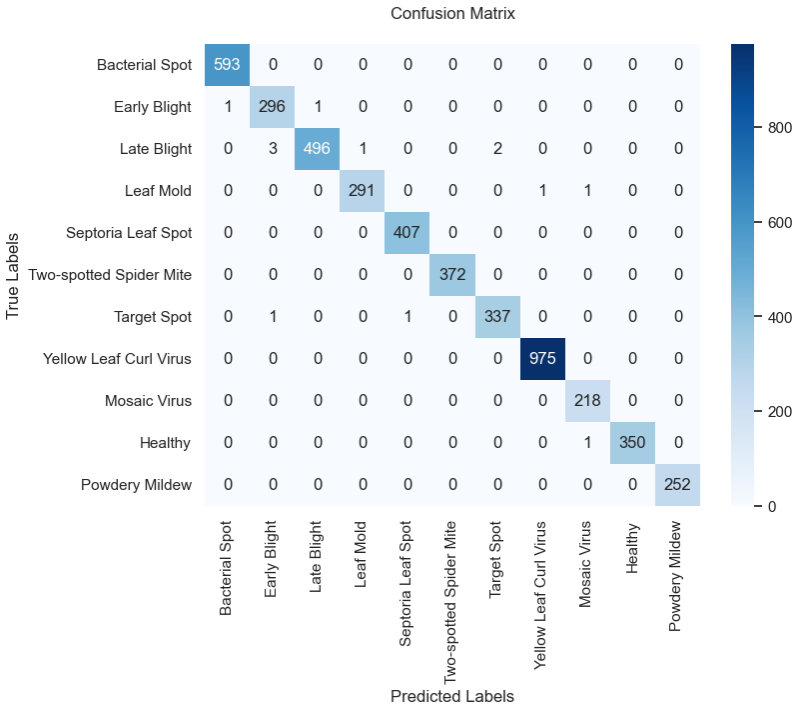

Fig. 8: Confusion Matrix of DenseNet-121 model

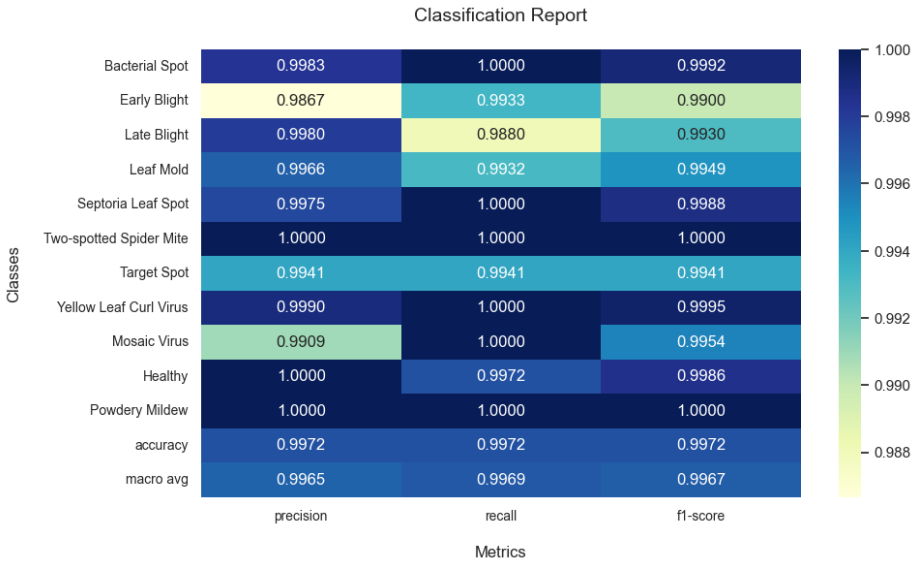

Fig. 9: Classification Report Heatmap of DenseNet-121 model

Table 4: Accuracy Comparison with other Deep Learning Models

Ref. No.	**Model Name**	Plant Species	Dataset	Accuracy (%)
[3]	Custom CNN	Tomato	PlantVillage	88.17
[10]	ResNet-34	14 Plant Species	PlantVillage	96.21
[9]	Custom CNN	Apple, Corn, Potato, Tomato	PlantVillage + Real-time	96.88
[8]	AlexNet, GoogleNet	Soybean	Soybean Field	98.75, 96.25
[2]	ResNet-101, AlexNet, VGG-16, GoogleNet, DenseNet-121	Tomato	PlantVillage	99.68, 97.85, 99.19, 98.52, 99.69
	(Proposed) DenseNet-121	Tomato	PlantVillage + TomatoLeaf	99.71 (highest)

4.1 Computational Feasibility

1. System Specifications used for training:
   - Processor: Intel Core i5-10300H (10th Gen, 4 cores, 8 threads).
   - GPU: NVIDIA GeForce GTX 1650 (4GB GDDR6 VRAM).
   - RAM: 16 GB DDR4.
   - Software Stack: TensorFlow 2.10, Keras API, Python 3.10.12.

2. Training Details:
   - Dataset size: 22,878 augmented images.
   - Training duration: 8 – 10 minutes per epoch.
   - Model achieved convergence at around 25^{th} epoch.
   - Inference time per image: < 50 milliseconds on GPU.

4.2 Deployment Challenges

1. Hardware Limitations: Mobile phones and edge hardware, such as the Raspberry Pi, might not be able to handle complete DenseNet-121 inference effectively.
2. Connectivity and Power Constraints: In rural places with poor connectivity, cloud-based inference may not be dependable.
3. Environmental Variability: Training data and real-world image circumstances are not the same like low light, leaves that are partially visible or covered, background noises like weeds, soil, or other crops.

5 Conclusion

We successfully utilized a DenseNset-121 deep learning model to detect diseases in tomato plants. The DenseNet-121 model demonstrated exceptional performance in detecting and identifying tomato plant diseases, achieving a peak training accuracy of 98.92% (49^{th} epoch) and a peak validation accuracy of 99.72% (34^{th} epoch). After evaluating the model on an independent validation (testing) dataset, having a total of 4600 images distributed in 11 different classes, the model impressively achieved an overall accuracy of **99.71%** in correctly identifying the diseases and classifying them to their respective classes, and giving the values of macro-precision, macro-recall, and F1-score as **99.65%**, **99.69%**, and **99.67%**, respectively; which are comparatively higher than any other deep learning models used previously on the same or similar datasets. These results clearly highlight the remarkable effectiveness and efficiency shown by the proposed model in recognizing tomato plant diseases.

6 Future Scope and Improvements

Overall, the results demonstrate the potential of DenseNet-121 model in tomato plant disease detection and classification, but there always exists room for improvements in the future. Let us briefly discuss such possible areas of improvements for disease detection deep learing/machine learning techniques.

- Transform the model into ONNX or TensorFlow Lite format and use this trained model for detecting the diseases in real time. This will help the farmers to save time.
- Test the model's performance in more practical and complex scenarios like raining, lightening, blurred images etc.

– Explore and experiment with other deep learning techniques by using hybrids of various models to generate new and more effective models which outperform DenseNet-121 alone.

By implementing these advancements and more, the DenseNet-121 model can further evolve into a robust tool and the disease detection process can become more useful for precision agriculture, and contributing significantly to sustainable farming practices improving the management of crop health.

References

1. Shanmugam, S. P., Murugan, M., Shanthi, M., Elaiyabharathi, T., Angappan, K., Karthikeyan, G., ... & Srinivasan, R.: Evaluation of Integrated Pest and Disease Management Combinations against Major Insect Pests and Diseases of Tomato in Tamil Nadu, India. Horticulturae 10(7), 766 (2024).
2. Gehlot, M., Saini, M. L.: Analysis of different CNN architectures for tomato leaf disease classification. In: 2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE), pp. 1-6. IEEE, December 2020.
3. Sakkarvarthi, G., Sathianesan, G. W., Murugan, V. S., Reddy, A. J., Jayagopal, P., Elsisi, M.: Detection and classification of tomato crop disease using convolutional neural network. Electronics 11(21), 3618 (2022).
4. LeCun, Y., Bengio, Y.: Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361(10), 1995.
5. Nagendran, K., Venkataravanappa, V., Chauhan, N. S., Kodandaram, M. H., Rai, A. B., Singh, B., Vennila, S.: Viral diseases: a threat for tomato cultivation in Indo-Gangetic eastern plains of India. Journal of Plant Pathology 101, 15-22 (2019).
6. Singh, V. K., Singh, A. K., Kumar, A.: Disease management of tomato through PGPB: current trends and future perspective. 3 Biotech 7, 1-10 (2017).
7. Thangaraj, R., Anandamurugan, S., Pandiyan, P., Kaliappan, V. K.: Artificial intelligence in tomato leaf disease detection: a comprehensive review and discussion. Journal of Plant Diseases and Protection 129(3), 469-488 (2022).
8. Jadhav, S. B., Udupi, V. R., Patil, S. B.: Identification of plant diseases using convolutional neural networks. International Journal of Information Technology 13(6), 2461-2470 (2021).
9. Gajjar, R., Gajjar, N., Thakor, V. J., Patel, N. P., Ruparelia, S.: Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform. The Visual Computer 1-16 (2022).
10. Chohan, M., Khan, A., Chohan, R., Katpar, S. H., Mahar, M. S.: Plant disease detection using deep learning. International Journal of Recent Technology and Engineering 9(1), 909-914 (2020).
11. Raghavendra, K. V., Balodi, R., Chander, S.: Integrated pest management approaches against major insect pests and diseases of tomato. Indian Horticulture 68(1), 39-42.
12. Gulati, A., Wardhan, H., Sharma, P.: Tomato, onion and potato (TOP) value chains. Agricultural Value Chains in India, 33 (2022).
13. Bhujade, V. G., Sambhe, V. K.: Multi-disease classification and severity estimation of cotton and soybean plants using DenseNet. In: International Conference on Advanced Network Technologies and Intelligent Computing, pp. 20-41. Springer, Cham (2023).