Plant Disease Detection Using CNN

Rakesh Vengala
Department of Computer Science and Engineering
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla, India
rakeshvengala8@gmail.com

Appala Swamy Satyavarapu
Department of Computer Science and Engineering
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla, India
swamysatyavarapu@gmail.com

Kethana Sai Pilli
Department of Computer Science and Engineering
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla, India
kethanasaipilli@gmail.com

Hussain Valli Yadati
Department of Computer Science and Engineering
Bapatla Engineering College
(Autonomous)
(Affiliated to Acharya Nagarjuna University)
Bapatla, India
yadatihussainvalli@gmail.com

***Abstract*—The modern agricultural landscape is plagued by various diseases threatening food security, caused by a range of pathogens like fungi, bacteria, and viruses. These diseases lead to significant crop yield reductions, economic losses, and disruptions in food supply chains. To combat these challenges, there's a need for advanced techniques in disease detection. Traditional methods often fall short, lacking accuracy and early detection capabilities. In response, deep learning methodologies, particularly Convolutional Neural Networks (CNNs), are being explored for automated plant leaf disease detection. CNNs excel in extracting intricate features from image data, making them suitable for categorizing diverse agricultural diseases. The primary aim of this research is to develop a method for detecting 38 distinct plant illnesses using simple techniques and minimal computational resources, surpassing the performance of standard models. This approach promises to revolutionize disease detection in agriculture, offering more accurate and timely interventions to safeguard crop health and food security.**

Keywords*— *Agriculture, CNN, Deep Learning, Plant Disease Detection, Automation

I. INTRODUCTION

India is an agricultural country, where most of the population depends on agricultural products. So the cultivation can be improved by technological support. Diseases may cause by pathogen in plant at any environmental condition. In most of the cases diseases are seen on the leaves of the plants, so the detection of disease plays an important role in successful cultivation of crops.

Diseases are very detrimental to the health of plants, and that in turn affects them development. India has made significant strides in pesticide, fungicide, and herbicide advancement and research. But each year, owing to unknown factors, plants fall to a variety of recognized illnesses, resulting in the loss of countless tons of yield. The assault of such different forms of crop diseases causes a significant reduction of crop yield both qualitatively and quantitatively.

Traditional methods for detecting diseases require manual inspection of plants by experts. This process needs to be continuous, and can be very expensive in large farms, or even completely unavailable to many small farm holders living in rural areas. This is why many attempts to automate disease detection have been made in the last few decades.

Current technological advancements have made the detection and diagnosis of plant diseases conceivable and achievable, hence paving the road for improved plant management in the event that a plant is infected. The suggested approach for the identification of plant leaf diseases concentrates on 14 plant species and 38 varieties or categories. Image Classification, Voice Recognition, and Processing of Natural Language has all shown exceptional performance over recent years due to Deep Learning. Utilizing a CNN to address the issue of identifying plant diseases yields excellent results.

CNN is acknowledged as the most effective Object Recognition technique. It is utilized for the creation of a predictive model that is operated on the input picture and changes the input in order to identify the output labels. In addition to classifying the plants, the model could also provide valuable information such as prevention measures and supplements. This could include recommendations for specific pesticides or fungicides to apply, cultural practices to reduce the risk of disease, or nutritional supplements to improve plant health.

II. DATASET

In this paper, the New Plant Diseases Dataset is used [8]. This dataset is recreated using offline augmentation from the original dataset which is plant village dataset. This dataset consists of about 87K RGB images of healthy and diseased crop leaves which is categorized into 38 different classes**.** A new directory containing 33 test images is created later for prediction purpose. The plants that are considered in this dataset are Orange, Grape, Raspberry, Tomato, Peach, Apple, Cherry, Soybean, Pepper bell, Corn, Potato, Blueberry, Squash and Strawberry. There are 17 fungal diseases, 4 bacterial diseases, 2 viral diseases, 2 mould diseases, and one mite disease present in this dataset.
This dataset consists of following directories:

1) Train (70295 images)
2) Test (33 images)
3) Valid (17572 images)

Figure 1.1 describes graph where The vertical bar plot illustrates the distribution of images in the training and validation directories across different classes of data. Each bar on the x-axis represents a specific class, while the height

of the bars corresponds to the number of images belonging to that class.

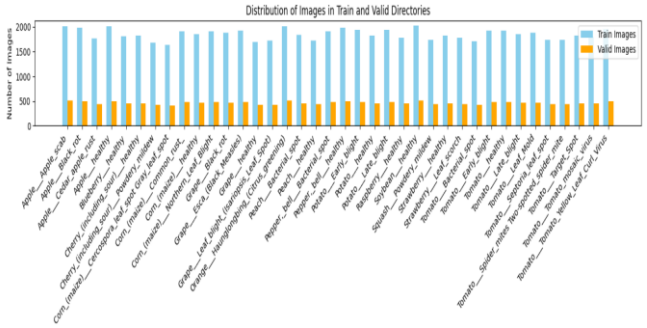

Figure 2.1 Number of images per class

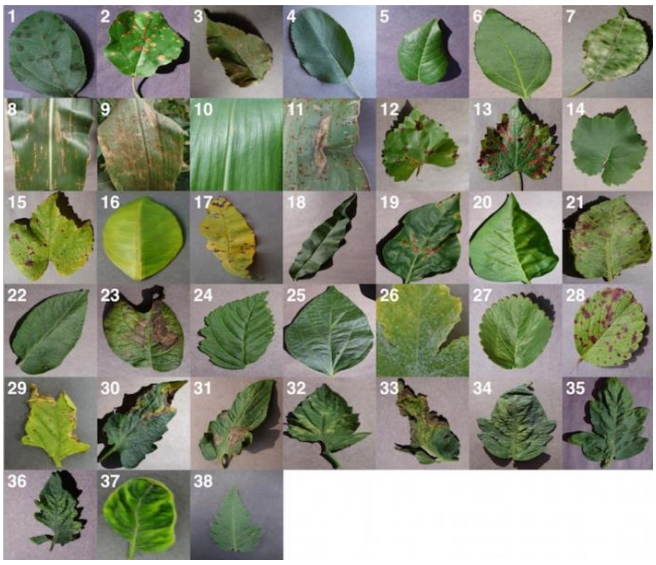

Figure 2.2 Example of each class in the dataset

III. Related Work

In this section, we provide the relevant background on previous work on Plant Disease Detection. Recently deep learning methods came into picture for accurate detection of plant diseases.

De Luna et al. [3] introduced an innovative approach for identifying diseases in tomato plants. They devised a motorized image capture container to photograph all four sides of each tomato crop, facilitating disease detection. The study focused on Diamante Max tomatoes and aimed to diagnose Phoma Rot, Leaf Miner, and Target Spot diseases. Data collection involved gathering both healthy and diseased leaves. Subsequently, a neural network utilizing deep convolution was constructed to recognize the three disorders. The system employed CNN to determine the prevalent tomato illnesses. The F-RCNN-based abnormality detection model achieved an 80% confidence score, while the Transfer Learning illness identification model demonstrated a precision of 95.75%.

V. Suma, R. A. Shetty, R. F. Tated, S. Rohan, and T. S. Pujar [4] present a CNN-based Leaf Disease Identification and Remedy Recommendation System. Emphasizing the critical role of farming in the economy, the study underscores the challenge farmers face in identifying plant diseases, leading to decreased agricultural output. The authors advocate for the use of recordings and photographs of leaves, providing agricultural experts with a more comprehensive perspective to address plant disease issues effectively. Leveraging technological advancements, the research proposes equipment capable of identifying and diagnosing crop illnesses, facilitating earlier recognition to mitigate their adverse effects on crops. The study focuses on utilizing image processing methods for crop disease identification, utilizing a dataset comprising 5000 photos of healthy and diseased plant leaves. Employing semi-supervised algorithms, the system aims to identify four types of diseases.

P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang [5] present a study on the Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolution Neural Networks. The research addresses five apple leaf diseases, namely brown spot, mosaic, aria leaf spot, rust, and grey spot, which significantly impact apple production. The study employs deep learning techniques to enhance Convolutional Neural Networks (CNNs) for detecting apple leaf diseases. Utilizing the apple leaf disease dataset (ALDD), they propose a novel apple leaf disease detection model, referred to as INAR-SSD, which utilizes deep-CNNs with the Google Net Inception structure and Rainbow concatenation. The INAR-SSD model is trained on a test dataset comprising 26,377 photos of apple leaf illnesses. Experimentally, the INAR-SSD model achieves a detection accuracy of 78.80%.

IV. Proposed Model

A. *Convolutional Neural Networks (CNN)*

Convolutional Neural Networks (CNNs) are a specialized type of Artificial Neural Network (ANN) designed primarily for processing matrix-like data. They are particularly effective in tasks related to computer vision, such as image recognition and classification. They are especially effective in tasks where the spatial relationships between data points matter, like recognizing patterns in images, text and time-series data as well. The complexity in terms of no. of parameters to be learned and the training time exponentially grows in case of ANNs.

CNNs consist of multiple layers-Convolution layers, Pooling layers and Fully connected layers.

Convolutional Layers: These layers apply a set of filters (small kernels/matrices) to the input data. Each filter learns to detect specific features (like edges or textures) within the data.

Pooling Layers: Pooling layers are often used in CNNs to reduce the spatial dimensions of the feature maps generated by the convolutional layers. Common pooling operations include max pooling and average pooling, which help in reducing computational complexity and controlling overfitting.

Fully Connected Layers: These are traditional densely connected neural network layers that process the features

learned by previous layers for tasks like classification. The flattened one-dimensional output of the pooling layers is fed to the first fully connected layer. These layers are often followed by a softmax activation function for classification tasks.

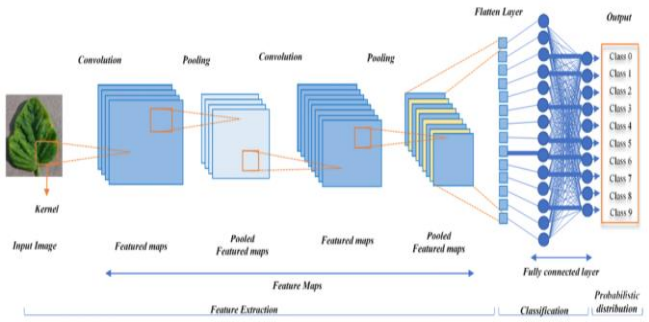

Figure 4.1 CNN Architecture

B. *Proposed Architecture*

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 32)      896

 conv2d_1 (Conv2D)           (None, 254, 254, 32)      9248

 max_pooling2d (MaxPooling2  (None, 127, 127, 32)      0
 D)

 conv2d_2 (Conv2D)           (None, 127, 127, 64)      18496

 conv2d_3 (Conv2D)           (None, 125, 125, 64)      36928

 max_pooling2d_1 (MaxPoolin  (None, 62, 62, 64)        0
 g2D)

 conv2d_4 (Conv2D)           (None, 62, 62, 128)       73856

 conv2d_5 (Conv2D)           (None, 60, 60, 128)       147584

 max_pooling2d_2 (MaxPoolin  (None, 30, 30, 128)       0
 g2D)

 conv2d_6 (Conv2D)           (None, 30, 30, 256)       295168

 conv2d_7 (Conv2D)           (None, 28, 28, 256)       590080

 max_pooling2d_3 (MaxPoolin  (None, 14, 14, 256)       0
 g2D)

 conv2d_8 (Conv2D)           (None, 14, 14, 512)       1180160

 conv2d_9 (Conv2D)           (None, 12, 12, 512)       2359808

 max_pooling2d_4 (MaxPoolin  (None, 6, 6, 512)         0
 g2D)

 dropout (Dropout)           (None, 6, 6, 512)         0

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 1500)              27649500

 dropout_1 (Dropout)         (None, 1500)              0

 dense_1 (Dense)             (None, 38)                57038

=================================================================
Total params: 32418762 (123.67 MB)
Trainable params: 32418762 (123.67 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 4.2 Proposed Architecture

The model begins with a series of Conv2D layers, each followed by a Rectified Linear Unit (ReLU) activation function, serving to extract hierarchical features from the input data. Subsequently, MaxPooling2D layers reduce the spatial dimensions of the feature maps, aiding computational efficiency and mitigating overfitting. Dropout layers are strategically incorporated to randomly deactivate a portion of neurons during training, promoting generalization and preventing overreliance on specific features. Following these convolutional and pooling layers, a Flatten layer reshapes the output into a one-dimensional array, facilitating the transition to the fully connected layers. The Dense layers at the end of the network connect every neuron in one layer to every neuron in the subsequent layer, culminating in an output layer with 38 neurons, likely corresponding to the number of classes in the classification task. The summary also provides insights into the model's complexity, revealing a total of 32,418,762 parameters, all of which are trainable, indicative of a sophisticated architecture capable of capturing intricate patterns in the data.

C. *Working Model*

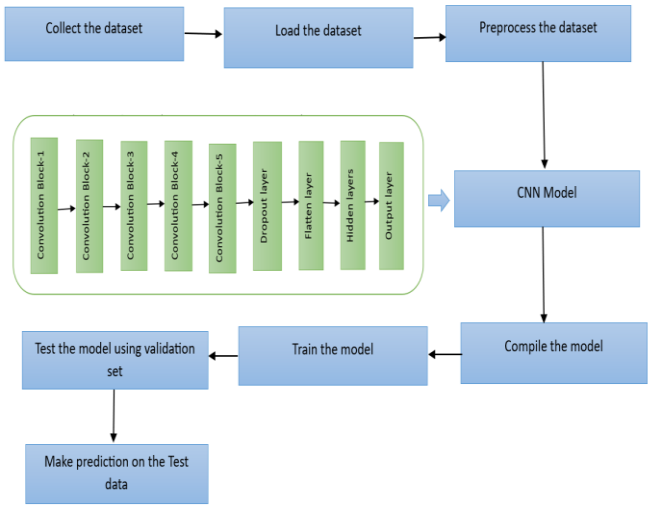

Figure 4.3 Working Flow of the Model

Collect the Dataset: This step involves gathering image data that will be used to train the CNN model. The dataset should be relevant to the classification task you intend to perform.

Load the Dataset: The collected dataset is loaded into the system for further processing.

Preprocess the Dataset: Preprocessing is crucial to ensure the data is consistent and suitable for training the model. Common preprocessing techniques for image data include resizing, batching, shuffling etc.

Define CNN Model Architecture: Here, we designed the CNN architecture, specifying the layers and connections that will process the image data and extract features. The architecture determines the model's capacity to learn complex patterns from the images. Common CNN layers include convolutional layers, pooling layers, activation functions, and fully connected layers.

*Compile the Model***:** This step involves configuring the training process by specifying the optimizer (algorithm used to adjust model weights) and the loss function (function

used to measure the model's prediction error). Common choices include optimizers like Adam or SGD (Stochastic Gradient Descent) and loss functions like categorical cross-entropy for multi-class classification problems.

Train the Model: The preprocessed data and compiled model are used for training. Training involves iterating through the dataset multiple times (epochs). In each iteration (batch), the model predicts outputs for a subset of images, calculates the loss based on the difference between predictions and true labels, and adjusts its internal weights using the optimizer to minimize the loss. This process helps the model learn to map the input images to their corresponding categories.

Test the Model using Validation Set: A validation set, separate from the training data, is used to monitor the model's performance during training. The model's accuracy (percentage of correct predictions) or other metrics like precision, recall, and F1-score are evaluated on the validation set.

Evaluate Model (Test Set): Once training is complete, the final model's performance is assessed using a separate test set. The test set is another unseen data partition used for a final evaluation of the model's generalizability on real-world data.

V. RESULTS

To evaluate the performance of the model, you would typically use a separate dataset (e.g., a validation set or a test set) that the model hasn't seen during training. This ensures an unbiased assessment of its generalization capabilities. The most common metrics for evaluating classification models include accuracy, precision, recall, F1-score.

Class Name	Precision	Recall	F1-Score
Apple___Apple_scab	0.99	0.97	0.98
Apple___Black_rot	0.97	0.99	0.98
Apple___Cedar_apple_rust	0.98	1.00	0.99
Apple___healthy	0.97	0.97	0.97
Blueberry___healthy	0.99	0.98	0.98
Cherry_(including_sour)___Powdery_mildew	0.98	1.00	0.99
Cherry_(including_sour)___healthy	0.97	0.99	0.98
Corn_(maize)___Cercospora_leaf_spot Gray_leaf	0.95	0.85	0.90
Corn_(maize)___Common_rust_	1.00	0.99	0.99
Corn_(maize)___healthy	1.00	0.98	0.99
Corn_(maize)___Northern_Leaf_Blight	0.87	0.97	0.92
Grape___Black_rot	0.97	1.00	0.98
Grape___Esca_(Black_Measles)	1.00	0.97	0.99
Grape___healthy	0.99	1.00	1.00
Grape___Leaf_blight_(Isariopsis_Leaf_Spot)	0.99	1.00	0.99
Orange___Haunglongbing_(Citrus_greening)	0.98	1.00	0.99
Peach___Bacterial_spot	0.97	0.98	0.97
Peach___healthy	0.99	1.00	0.99
Pepper,_bell___Bacterial_spot	0.99	0.97	0.98
Pepper,_bell___healthy	1.00	0.94	0.97
Potato___Early_blight	0.94	1.00	0.97
Potato___healthy	0.99	0.98	0.99
Potato___Late_blight	0.92	0.99	0.95
Raspberry___healthy	0.98	1.00	0.99
Soybean___healthy	0.96	0.99	0.98
Squash___Powdery_mildew	0.98	0.99	0.99
Strawberry___healthy	1.00	1.00	1.00
Strawberry___Leaf_scorch	0.99	1.00	0.99
Tomato___Bacterial_spot	0.97	0.97	0.97
Tomato___Early_blight	0.95	0.90	0.92
Tomato___healthy	0.99	1.00	0.99
Tomato___Late_blight	0.96	0.83	0.89
Tomato___Leaf_Mold	0.99	0.99	0.99
Tomato___Septoria_leaf_spot	0.96	0.89	0.93
Tomato___Spider_mites Two-spotted_spider_mite	0.98	0.96	0.97
Tomato___Target_Spot	0.93	0.97	0.95
Tomato___Tomato_mosaic_virus	0.98	1.00	0.99
Tomato___Tomato_Yellow_Leaf_Curl_Virus	0.99	0.99	0.99

Table 5.1 Performance Evaluation of crop diseases

Figure 5.2 and Figure 5.3 is showing training & validation accuracy and training & validation loss graph respectively. The objective of the "Plant Disease Detection Using CNN" research is to develop a neural network competent of recognizing 14 crop species and 26 prevalent illnesses. When verified in a controlled setting, an overall accuracy of 97.34% is shown.

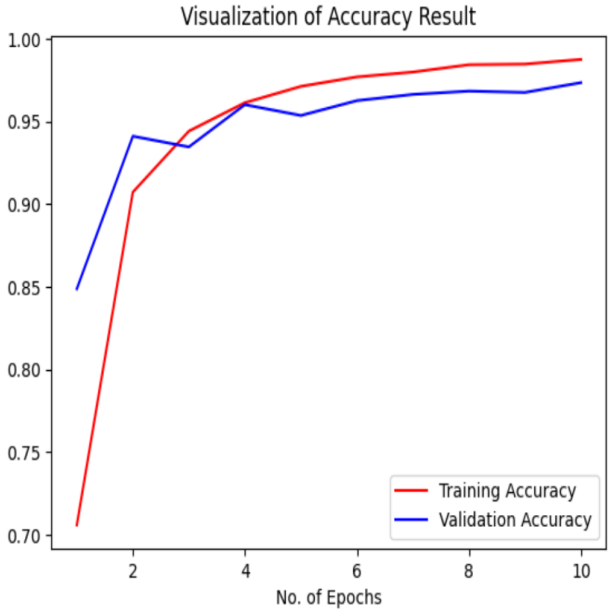

Figure 5.1 Training and Validation Accuracy Graph

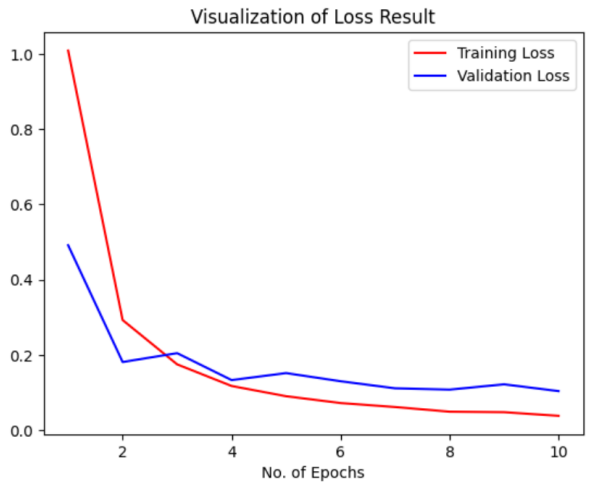

Figure 5.2 Training and Validation Loss Graph

VI. Conclusion

The proposed algorithm is implemented successfully to train the system. The accuracy percentage on the test set is 97.34% with no overfitting. We hope that the results of this paper will encourage future work including Real-Time processing enhancements, Integration with IoT and Sensor Data and also this work can expanded into further to develop an app through which one would also know the remedy to a plant disease.

Acknowledgment

The authors would like to thank Ass. Prof. Mr. Krishna Kishore Thota, Bapatla Engineering College, Bapatla for guiding throughout the work and the authors would also thank research paper writers for providing base to our work.

References

[1] N. M. Madhav and U. Scholar, "Plant Disease Detection Using Deep Learning," International Research Journal on Advanced Science Hub (IRJASH) Special, vol. 03, 2021, Accessed: Nov. 25, 2022. [Online]. Available: www.rspsciencehub.

[2] L. Poole, D. B.- Intelligence, B. Data, C. and Data, and undefined 2021, "Investigating popular CNN architectures for plant disease detection,"ieeexplore.ieee.org,doi:10.1109/icABCD51485.2021.9519341.

[3] R. G. de Luna, E. P. Dadios, and A. A. Bandala, "Automated Image Capturing System for Deep Learning based Tomato Plant Leaf Disease Detection and Recognition," IEEE Region 10 Annual International Conference, Proceedings/TENCON, vol. 2018-October, pp. 1414–1419, Feb. 2019, doi:10.1109/TENCON.2018.8650088.

[4] V. Suma, R. A. Shetty, R. F. Tated, S. Rohan, and T. S. Pujar, "CNN based Leaf Disease Identification and Remedy Recommendation System," Proceedings of the 3rd International Conference on Electronics and Communication and Aerospace Technology, ICECA 2019, pp. 395– 399, Jun. 2019, doi: 10.1109/ICECA.2019.8821872.

[5] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, "Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks," IEEE Access, vol. 7, pp. 59069–59080, 2019, doi: 10.1109/ACCESS.2019.2914929.

[6] G. Geetharamani and A. P. J., "Identification of plant leaf diseases using a nine-layer deep convolutional neural network," Computers & Electrical Engineering, vol. 76, pp. 323–338, Jun. 2019, doi:10.1016/J.COMPELECENG.2019.04.011.

[7] P. Soni and R. Chahar, "A segmentation improved robust PNN model for disease identification in different leaf images," 1st IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems, ICPEICES 2016, Feb. 2017, doi:10.1109/ICPEICES.2016.7853301.

[8] "New Plant Diseases Dataset | Kaggle." https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset

[9] X.-P. Fan, J.-P. Zhou, and Y. Xu, ''Recognition of field maize leaf diseases based on improved regional convolutional neural network,'' J. South China Agricult. Univ., vol. 41, no. 6, pp. 82–91, Jun. 2020.

[10] T. Subetha, R. Khilar, M. C.-M. T. Proceedings, and undefined 2021, "A comparative analysis on plant pathology classification using deep learning architecture–Resnet and VGG19," Elsevier.

[11] S. Vallabhajosyula, V. Sistla, V. K.-J. of P. D. and, and undefined 2022, "Transfer learningbased deep ensemble neural network for plant leaf disease detection," Springer.