iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 Issue: VII Month of publication: July 2025

DOI: https://doi.org/10.22214/ijraset.2025.73107

Plant Disease Detection Using Deep Learning

Jaya Kethan Reddy Dulla[1], M. Dhanalakshmi[2]
[1]*Post-Graduate Student, Department of Information Technology, Data Science, Jawaharlal Nehru Technological University, Hyderabad, India*
[2]*Professor of Information Technology, Jawaharlal Nehru Technological University, Hyderabad, India*

Abstract: The early and accurate detection of plant diseases plays a pivotal role in enhancing crop health and ensuring food security in modern agriculture. Traditional disease diagnosis techniques often rely on visual inspection by experts, which may be subjective, time-consuming, and inaccessible to farmers in remote regions. To overcome these limitations, this project introduces an intelligent plant disease detection system that leverages deep learning and image analysis to identify symptoms directly from leaf images. The proposed methodology begins with the collection of a curated dataset comprising various plant species exhibiting both healthy and diseased conditions. The images undergo preprocessing to enhance quality and ensure consistency, followed by feature extraction using Convolutional Neural Networks (CNNs). Transfer learning is applied to improve model generalization and reduce the training time by utilizing pre-trained models. The system is integrated into a Flask-based web application, enabling users to upload leaf images and receive instant disease diagnoses along with treatment suggestions and suitable fertilizers. Evaluation of the model has shown high classification accuracy across multiple disease classes, affirming its potential to support precision agriculture. The solution is designed to be lightweight, user-friendly, and deployable in real-world agricultural settings, aiming to assist farmers with timely and informed interventions to mitigate crop loss and promote sustainable farming practices.

Keywords: CNN, Plant Disease Detection, Deep Learning, Image Processing, Flask Web App, Transfer Learning, Agriculture, Real-time Diagnosis.

I. INTRODUCTION

Agriculture plays a crucial role in India's economic development, with nearly 70% of the population depending on it either directly or indirectly. Given its significance, any decline in crop productivity due to plant diseases can lead to considerable economic setbacks. Since plant leaves often exhibit the earliest signs of disease, they serve as key indicators for early diagnosis. Continuous monitoring of crops from their germination stage to harvest is essential to mitigate the impact of diseases. Traditionally, disease detection relied on visual inspection by experts—a manual, time-consuming, and labour-intensive method that is not scalable for large-scale farming. To address this challenge, our project introduces a deep learning-based system that leverages Convolutional Neural Networks (CNNs) to detect diseases in plant leaves, particularly focusing on crops like cotton. This automated solution facilitates early diagnosis of bacterial infections, reducing the need for farmers to be physically present in the field for prolonged periods. Early and accurate disease identification is crucial, as it supports timely treatment and minimizes damage to both crop yield and quality. Additionally, the system empowers farmers by providing reliable diagnoses without requiring advanced technical expertise. In recent years, intelligent plant disease detection techniques have gained traction, offering faster, cost-effective, and more accurate alternatives to traditional methods. These AI-powered solutions eliminate the dependency on manual observation and reduce the likelihood of human error in disease recognition. Our model is trained to recognize the subtle visual symptoms that often go unnoticed by the naked eye, such as changes in leaf color, texture, and shape. By automating the diagnosis process, this system promotes more sustainable agricultural practices and supports food security by minimizing yield loss due to undetected plant diseases.

A. Objective

1) To build an automated system that takes audio input and turns it into a complete, high-quality video, helping reduce manual work and speed up the content creation process.
2) To make use of powerful AI tools—like Whisper for converting speech to text, Groq for generating meaningful scripts, Pexels API for finding suitable images, and ESRGAN for improving video quality—to create visually engaging and informative content.
3) To ensure the final video feels immersive by properly syncing audio with relevant visuals, offering a smooth and impactful viewing experience.

4) To provide an easy and innovative way for creators, educators, and media professionals to generate videos directly from audio, making the content creation process faster and more accessible.

II. LITERATURE SURVEY

1) Brahimi et al. presented a CNN-based system for identifying plant leaf diseases. The model worked well for common crops, but its performance was constrained by the size and quality of the training dataset. Their work showed promising accuracy but lacked scalability across diverse plant types.
2) Sladojevic et al. implemented a deep learning-based approach using CNN for feature extraction. The system demonstrated high accuracy in disease classification but was resource-intensive, requiring longer training times and access to powerful hardware like GPUs.
3) Mohanty et al. used transfer learning by applying models like AlexNet and VGG trained on large image datasets. This method significantly improved efficiency and reduced training time, though the system's success heavily depended on having well-labelled and clean image data.
4) Kamilaris and Prenafeta-Boldú developed a real-time, mobile-compatible deep learning model that allowed users to detect plant diseases on-site using a smartphone. While convenient and scalable, their app's accuracy dropped under poor lighting or image quality conditions.
5) Ferentinos trained CNNs on a very large dataset containing more than 80,000 images across 58 different classes of plant diseases and healthy crops. This approach achieved very high accuracy but faced challenges when applied to new or uncommon plant diseases do not present in the training set.

III.METHODOLOGY OF THE PROPOSED SYSTEM

A. Proposed System

The proposed system follows a systematic pipeline that includes data collection, preprocessing, feature extraction, classification, and web-based deployment. Each step is crucial in building a robust and accurate disease detection model using deep learning techniques. The methodology is designed to ensure high performance in both model accuracy and real-time usability.

1) Data Collection

A high-quality image dataset is critical for training a deep learning model effectively. This project uses the **PlantVillage dataset**, a well-known open-source repository that contains labeled images of healthy and diseased plant leaves. The dataset includes multiple classes covering crops like tomato, potato, and bell pepper with various disease categories. Each image is labelled for supervised learning and represents diverse real-world conditions.

2) Image Preprocessing

To ensure consistency and improve learning performance, preprocessing steps are applied to the raw input images:

- Resizing: All images are resized to a standard dimension of 256x256 pixels.
- Normalization: Pixel values are scaled between 0 and 1 to ensure numerical stability during training.
- Augmentation: Techniques like rotation, flipping, and zooming are applied to artificially expand the dataset and improve generalization.
- Color Standardization: Images are converted to RGB format for consistent channel interpretation.
- Feature Extraction using CNN: Convolutional Neural Network (CNN) is used to extract hierarchical image features. CNN is well-suited for spatial data like images and excels in detecting edges, textures, and patterns which are essential for identifying symptoms of plant diseases. The model architecture includes:

Convolutional Layers to extract low-level features
ReLU Activation to introduce non-linearity
Pooling Layers to reduce spatial dimensions
Fully Connected Layers for classification
Transfer learning is applied using a pre-trained model (e.g., ResNet50 or VGG16) to reduce training time and improve performance.

3) Model Training

The model is trained using the pre-processed image data with categorical cross-entropy loss and the Adam optimizer. The training process involves:

Dividing the dataset into training and testing subsets.

Monitoring performance through metrics like accuracy, precision, recall, and F1-score.

Using early stopping to prevent overfitting and save the best performing model.

Real-Time Prediction via Flask Web Interface

To make the model accessible to end-users, a Flask-based web application is developed. This allows users to upload a leaf image and receive instant disease diagnosis. The pipeline includes:

Accepting the uploaded image.

Preprocessing it similarly to the training pipeline.

Predicting the class using the trained CNN.

Displaying disease name, plant type, and treatment recommendations.

B. System Architecture

The system architecture defines the overall structure and workflow of the plant disease detection model. It outlines how various components interact to process input leaf images, extract meaningful features, and classify them based on health conditions. This design integrates both feature compression and deep learning techniques to ensure accurate and efficient disease identification. By combining a Convolutional Autoencoder (CAE) for dimensionality reduction and a Convolutional Neural Network (CNN) for classification, the system achieves a balance between performance and computational efficiency. The architecture also supports a structured flow for both training and testing phases, ensuring consistent results across different image inputs.

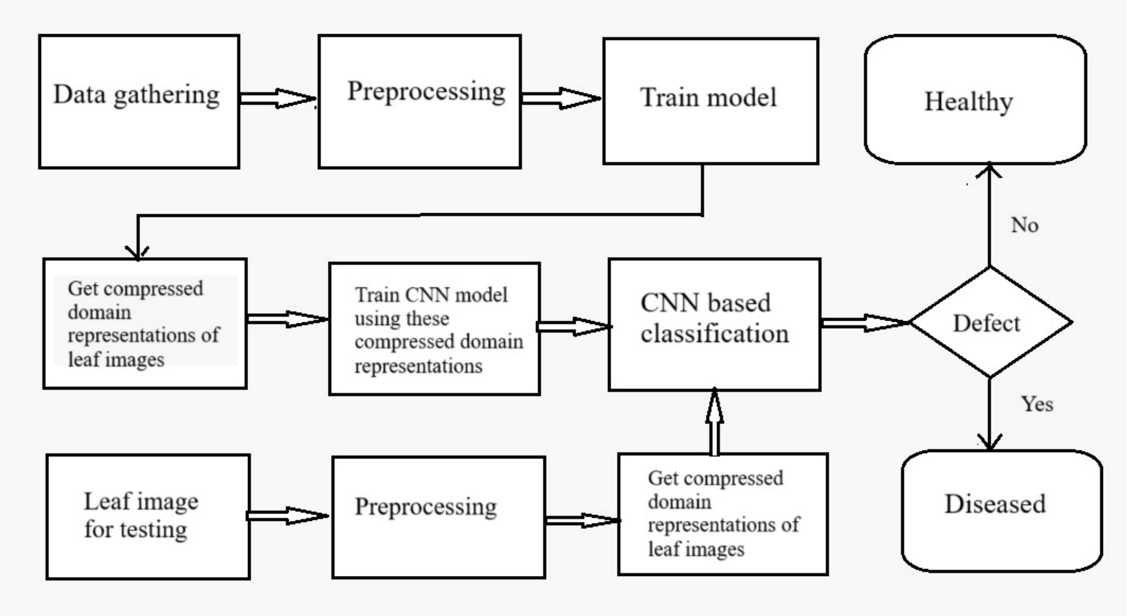

Data gathering
Preprocessing
Train model
Healthy
No
Get compressed domain representations of leaf images
Train CNN model using these compressed domain representations
CNN based classification
Defect
Yes
Leaf image for testing
Preprocessing
Get compressed domain representations of leaf images
Diseased

Fig: System Architecture

1) Data Collection: The initial step involves gathering a comprehensive set of plant leaf images from various crops. This dataset contains examples of both healthy and infected leaves, each displaying different visual symptoms. A diverse and well-balanced dataset is essential for training a reliable model, as it allows the system to learn from a wide range of disease conditions and improves its ability to handle unseen test images.
2) Image Preprocessing: Before any analysis begins, the raw images undergo preprocessing to prepare them for model input. This includes resizing each image to a fixed dimension (commonly 256×256 pixels), normalizing pixel values, converting color formats, and enhancing contrast where needed. These steps ensure consistency in input format and eliminate noise or distortions, resulting in better training performance.

3) Feature Reduction Using Autoencoders: After preprocessing, the images are passed through a Convolutional Autoencoder (CAE). This specialized neural network compresses each image into a lower-dimensional feature vector while retaining important visual characteristics. The encoder part of the CAE captures only the most meaningful data, helping reduce storage and processing demands without sacrificing accuracy.
4) CNN Model Training: The compressed outputs from the encoder are then used to train a Convolutional Neural Network (CNN). The CNN is responsible for learning to differentiate between various plant diseases and healthy leaves. Since the input data is already simplified through compression, the CNN can train more efficiently and focus on recognizing disease-specific textures and patterns.
5) Test Image Input via Web Interface: In the practical application phase, users such as farmers upload new leaf images through a user-friendly web interface. The system then follows the same pipeline that was used during training to analyze the new image.
6) Preprocessing of Test Images: The uploaded image is subjected to the same preprocessing operations applied during training. This ensures the image has the correct format, resolution, and quality before feature extraction begins, which is crucial for consistent results.
7) Feature Extraction from Test Images: Using the pre-trained encoder component of the CAE, the system generates a compact set of features from the test image. These features summarize the image's content—such as shape, edges, and color patterns—that are most relevant for detecting signs of disease.
8) Classification using CNN: Finally, the extracted features are passed to the trained CNN, which predicts whether the leaf is healthy or affected by a disease. If a disease is detected, the system may further specify the type of infection. The classification decision is based on the network’s prior learning and pattern recognition capabilities.

C. Algorithm

The Convolutional Neural Network (CNN) algorithm used for plant disease detection follows a systematic sequence of operations to process input images and classify them accurately. The process begins with the input layer, where an image of a plant leaf is taken and resized to a consistent dimension, typically 256×256 pixels. This image is then passed into the convolutional layers, where multiple filters slide over the image and extract essential features such as color changes, spots, edges, or textures that are commonly associated with plant diseases. After each convolution operation, the ReLU (Rectified Linear Unit) activation function is applied to introduce non-linearity, ensuring the model can capture complex patterns in the data. Next, the data is passed through a pooling layer, usually max pooling, which reduces the spatial size of the feature maps. This helps in decreasing computation time and making the model less sensitive to small distortions in the input. The output of the pooling layer is then flattened into a one-dimensional vector to prepare it for the dense layers. This flattened feature vector is fed into one or more fully connected layers, where the model combines all the extracted features and learns to classify the image based on disease characteristics. Finally, the data reaches the output layer, which applies a softmax activation function to generate a probability distribution over all possible classes. The class with the highest probability is selected as the prediction — for example, “Potato – Late Blight” or “Tomato – Healthy.” This structured approach allows CNNs to effectively analyze plant leaf images and identify diseases with high accuracy.

IV. IMPLEMENTATION AND RESULTS

A. Implementation Steps

1) Image Upload via Web Interface: The process begins when the user—typically a farmer or agronomist—accesses the Flask-based web application. The interface allows the user to upload an image of a plant leaf, which could either be healthy or affected by a disease. This feature ensures ease of use and accessibility across devices
2) Image Storage and Path Retrieval: Once the image is uploaded, it is securely stored in a predefined local directory on the server (e.g., /uploads). The system then retrieves the full file path of the image, which will be used in subsequent processing stages.
3) Image Resizing and Normalization (Preprocessing): To make the uploaded image compatible with the CNN model’s expected input format, it is resized to 256x256 pixels. The resizing ensures that the model can process different image dimensions consistently. The pixel values are also scaled (typically divided by 255) so that the data falls within a normalized range (0 to 1), which improves model performance and accuracy.
4) Real-Time Prediction Pipeline: The preprocessed image is then passed into the pre-trained Convolutional Neural Network (CNN). This model has already been trained on a labeled dataset of plant leaf images—both healthy and diseased. It analyzes the features in the uploaded image and classifies it into one of the predefined categories.

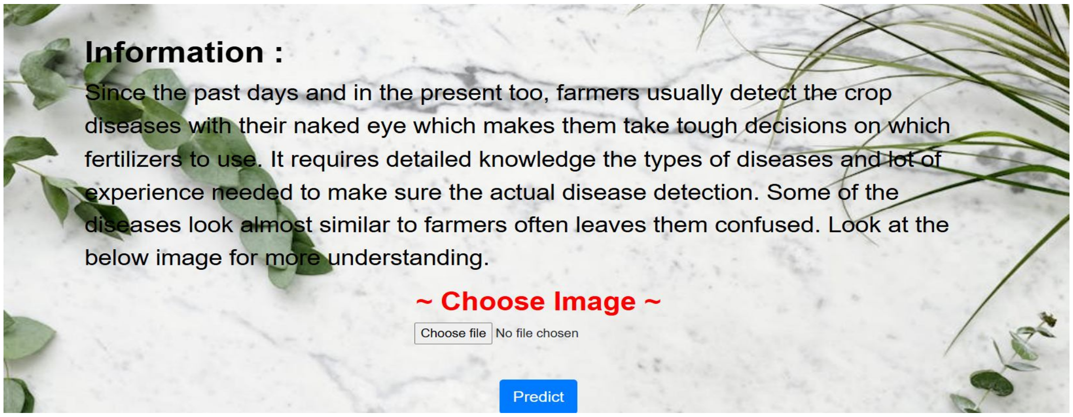
5) Diagnosis and Feature Matching: The CNN processes the image through multiple internal layers (like convolution, activation, pooling, and fully connected layers) to extract and match features such as color changes, texture anomalies, or spots that indicate disease. It then predicts the plant name and identifies the disease, if any.
6) Intelligent Output Generation: Based on the classification result (using the model's highest-probability output), the system dynamically generates a diagnosis report. This includes:

- The name of the plant.
- The detected disease or an indication that the plant is healthy.
- Suggested treatment methods.
- Recommended fertilizers or fungicides (if needed).

7) Result Rendering and Display: All this information is passed to the result page (result.html) and displayed to the user in an easily readable format. The entire flow—from image upload to output rendering—is executed in real time, typically taking just a few seconds.
8) Model Maintenance and Updates: The underlying model can be periodically updated with new data collected from real-world usage. By retraining with an expanded dataset, the system improves its ability to detect a wider range of diseases and adapt to different environmental conditions.

B. *Results*

It highlights how effectively the system identifies various plant diseases using deep learning techniques, particularly Convolutional Neural Networks (CNN). The evaluation is based on test images, comparing the system's predictions with actual classifications to measure accuracy and reliability. The results demonstrate the practicality and efficiency of the model in real-time disease identification, providing valuable insights into its effectiveness and potential for real-world agricultural use.



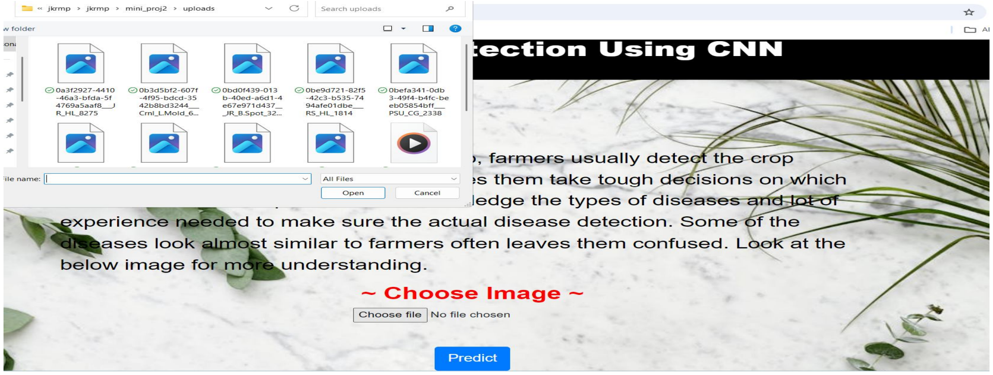
Information :

Since the past days and in the present too, farmers usually detect the crop diseases with their naked eye which makes them take tough decisions on which fertilizers to use. It requires detailed knowledge the types of diseases and lot of experience needed to make sure the actual disease detection. Some of the diseases look almost similar to farmers often leaves them confused. Look at the below image for more understanding.

~ Choose Image ~

Fig: User Interface

It shows the User Interface to choose and upload the image from your device.



« jkrmp › jkrmp › mini_proj2 › uploads
Search uploads
w folder
0a3f2927-4410-46a3-bfda-5f4769a5aaf8___JR_HL_8275
0b3d5bf2-607f-4f95-bdcd-3542b8bd3244___Crnl_L.Mold_6...
0bd0f439-013b-40ed-a6d1-4e67e971d437___JR_B.Spot_32...
0be9d721-82f5-42c3-b535-7494afe01dbe___RS_HL_1814
0befa341-0db3-49f4-b4fc-beeb05854bff___PSU_CG_2338
File name:
All Files
Open
Cancel
tection Using CNN

), farmers usually detect the crop
es them take tough decisions on which
ledge the types of diseases and lot of
experience needed to make sure the actual disease detection. Some of the diseases look almost similar to farmers often leaves them confused. Look at the below image for more understanding.

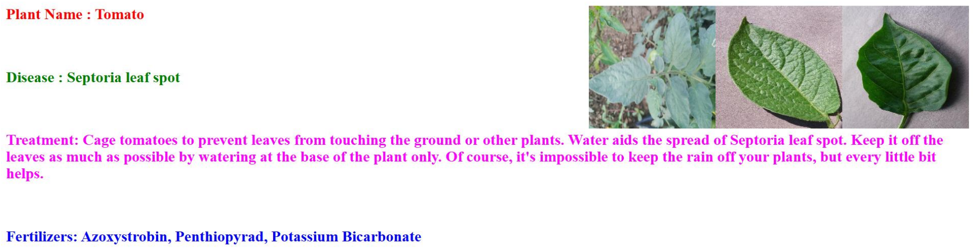
~ Choose Image ~

Choose file
No file chosen
Predict

Fig: Uploading images

In fig: We must choose one image. By clicking on predict the output will be displayed.

OUTPUT



Plant Name : Tomato

Disease : Septoria leaf spot

Treatment: Cage tomatoes to prevent leaves from touching the ground or other plants. Water aids the spread of Septoria leaf spot. Keep it off the leaves as much as possible by watering at the base of the plant only. Of course, it's impossible to keep the rain off your plants, but every little bit helps.

Fertilizers: Azoxystrobin, Penthiopyrad, Potassium Bicarbonate

Fig: Detecting Disease

V. LIMITATIONS AND FUTURE SCOPE

Although the proposed plant disease detection system using deep learning shows promising accuracy and efficiency, it does come with a few limitations. The performance of the model is heavily dependent on the quality and size of the dataset. In real-world environments, varying lighting conditions, complex backgrounds, and overlapping leaves can impact the accuracy of predictions. Additionally, the model might struggle with identifying diseases in their very early stages if symptoms are not visually prominent. Another limitation is that the model is typically trained on a fixed set of plant species and diseases; thus, its ability to generalize to new or rare cases is limited. Moreover, reliance on high computational resources may restrict deployment on devices with lower processing capabilities such as mobile phones without optimization. Looking ahead, the system can be expanded to include real-time mobile applications using optimized lightweight models like MobileNet or TensorFlow Lite. Increasing the dataset size with diverse environmental conditions and disease stages can help improve robustness. Integration with GPS and IoT devices can make it more useful for smart farming applications. Moreover, by incorporating multi-modal data like soil condition, humidity, and temperature, the model can offer more accurate and context-aware predictions. Future versions can also support automated treatment suggestions, notifications to local agricultural officers, or integration with drones for large-scale farm monitoring. These improvements would significantly enhance the utility and scalability of the system for modern agriculture.

VI. CONCLUSION

The proposed system effectively demonstrates how deep learning techniques, particularly Convolutional Neural Networks (CNNs), can be applied to identify plant diseases with high accuracy using leaf images. By automating the detection process, the model helps in early diagnosis, which is crucial in minimizing crop damage and improving agricultural productivity. Through systematic stages such as data collection, preprocessing, feature extraction, and classification, the system is capable of distinguishing between healthy and diseased plants and even identifying specific types of diseases. The integration of a Flask-based web application further enhances the usability of the system by offering real-time predictions to end-users like farmers or agronomists. Although certain limitations exist—such as environmental noise or limited training data—the project lays a solid foundation for building intelligent, scalable, and practical tools for plant health monitoring. Overall, this work highlights the potential of AI-driven solutions in advancing smart agriculture and promoting sustainable farming practices.

VII. ACKNOWLEDGMENT

I sincerely thank Sri M. Dhanalakshmi madam for her continual support in helping me carry out this project. Also extend my heartfelt gratitude to the authors of the research papers reviewed in this study.

REFERENCES

[1] M. H. Saleem, J. Potgieter, and K. M. Arif, "Plant Disease Detection and Classification by Deep Learning," *Plants*, vol. 8, no. 11, Art. 468, Nov. 2019. PubMed+11MDPI+11PMC+11

[2] S. Mohanty, D. P. Hughes, and M. Salathé, "Using Deep Learning for Image-Based Plant Disease Detection," *Frontiers in Plant Science*, vol. 7, p. 1419, 2016. arXiv+1Wikipedia+1

[3] L. Li, S. Zhang, and B. Wang, "Plant Disease Detection and Classification by Deep Learning—A Review," *IEEE Access*, vol. 2021, Art. No. 3069646, 2021. Astrophysics Data System+1ResearchGate+1

[4] Z. Zainab et al., "Plant Disease Detection Using Deep Learning Techniques," *IECE Journal of Image Analysis and Processing*, vol. 1, no. 1, pp. 36–44, Mar. 2025.

[5] B. S. Hamed, M. M. Hussein, and A. M. Mousa, "Plant Disease Detection Using Deep Learning," *International Journal of Intelligent Systems and Applications (IJISA)*, vol. 15, no. 6, pp. 38–50, Dec. 2023