International Journal of Current Research and Techniques (IJCRT)
E-ISSN: 2349-3194 (Online) | P-ISSN: 2348-4446 (Print)
UGC Approved Journal No: 47722 [No: 2616]
DOI: https://doi.org/10.61359/2024050044

# Plant Lead Disease Detection Using Machine Learning

**Dr. Poornima Raikar,** Assistant Professor, KLS VDIT Haliyal, Karnataka, India

**Ashish S,** Student, KLS VDIT Haliyal, Karnataka, India

*Manuscript Received: Jan 01, 2025; Revised: Jan 8, 2025; Published: Jan 10, 2025.*

**Abstract**

Identifying plant diseases visually is a labor-intensive task, less accurate, and limited to specific areas. However, using an automatic detection technique requires less effort, saves time, and achieves higher accuracy. Common plant diseases include brown and yellow spots, early and late scorch, as well as fungal, viral, and bacterial infections. Image processing is used to measure the affected area and to detect color differences in the diseased regions. Therefore, image processing plays a vital role in the detection of plant diseases. Disease detection involves steps such as image acquisition, image preprocessing, image segmentation, feature extraction, and classification. This project focuses on methods for detecting plant diseases using images of their leaves.

**Keywords:** Neural Networks, CNN, Image Processing, Classification

## 1. Introduction

India is a rapidly developing country, and agriculture has been the backbone of its development in the early stages. However, due to industrialization and globalization, the agricultural sector is facing numerous challenges. Additionally, there is a need to instill awareness and emphasize the importance of cultivation in the minds of the younger generation.

Nowadays, technology plays a vital role across all fields, but traditional methodologies are still being used in agriculture. Misidentifying plant diseases can lead to significant losses in yield, time, money, and product quality. Identifying the condition of plants is crucial for successful cultivation. In the past, this was done manually by experienced individuals, but environmental changes have made predictions increasingly difficult. Image processing techniques offer a solution for identifying plant diseases more effectively.

Plant disease symptoms can generally be observed on leaves, stems, flowers, etc. Here, we focus on using leaf images to identify diseases affecting plants. Agriculture today is more than just a food source; it also plays a critical role in the Indian economy, which is highly dependent on agricultural productivity. Therefore, in the field of agriculture, detecting plant diseases is essential. Early detection, facilitated by automatic disease detection techniques, can be highly beneficial, particularly in preventing losses.

Currently, plant disease detection is primarily done through naked-eye observation by experts. This method requires a large team of specialists and continuous monitoring, making it expensive and impractical for large farms. Additionally, in some countries, farmers lack proper facilities or awareness of how to contact experts, further increasing the costs and time associated with consulting professionals. In such cases, the proposed technique of automatic disease detection proves advantageous for monitoring large fields of crops. Detecting plant diseases by analyzing symptoms on leaves becomes more efficient, cost-effective, and accessible with this approach.

## 2. Literature Survey

Savita N. Ghaiwat et al. presented a survey on different classification techniques that can be used for plant leaf disease classification. Among these, the k-nearest neighbor (KNN) method appears to be both suitable and the simplest algorithm for class prediction. However, one drawback of Support Vector Machines (SVMs) is that if the training data is not linearly separable, determining optimal parameters becomes challenging.

The paper by Mrunalini et al. introduces a technique to classify and identify various diseases affecting plants. For automatic detection of diseases in leaves, neural networks are employed. This approach supports accurate detection of leaf diseases and proves significant for identifying diseases in stems and roots with minimal computational effort.

Kulkarni et al. proposed a methodology for early and accurate plant disease detection using artificial neural networks (ANN) and various image processing techniques. Their approach, which utilizes an ANN classifier for classification and a Gabor filter for feature extraction, achieves a recognition rate of up to 91%. The ANN-based classifier identifies different plant diseases by combining textures, colors, and features for disease recognition.

Sachin D. Khirade et al. proposed a methodology for detecting plant diseases using leaf images. Their system discusses various techniques for segmenting the diseased parts of plants. This methodology also incorporates feature extraction and classification techniques to analyze the features of infected leaves and classify plant diseases accurately. Precise detection and classification of plant diseases are crucial for successful crop cultivation, which can be achieved using image processing. They proposed using ANN methods for plant disease classification, such as self-organizing feature maps, backpropagation algorithms, and SVMs.

. Malathi and K. Aruli et al. presented a survey on plant leaf disease detection using image processing techniques. Diseases in crops cause significant reductions in the quantity and quality of agricultural products. Identifying disease symptoms with the naked eye is often challenging for farmers. Crop protection, especially on large farms, can benefit from computerized image processing techniques, which detect diseased leaves using color information. Depending on the application, many image processing techniques have been introduced to address these challenges using pattern recognition and automatic classification tools.

## 3.Methodology

We use the Convolutional Neural Network (CNN) algorithm for the project implementation. CNN is a class of deep learning neural networks that has revolutionized image recognition. These networks are commonly used to analyze visual imagery and are often employed in image classification tasks. CNNs are integral to applications ranging from Facebook's photo tagging to self-driving cars and are also widely used in healthcare and security.

In this project, digital cameras or similar devices are used to capture images of leaves, which are then analyzed to identify affected areas. Various image processing techniques are applied to these images to extract useful features required for analysis. Our current model is capable of predicting **Cercospora leaf spot disease**, **Common rust disease** in maize plants, and **Bacterial spot disease** in peach plants.

The proposed methodology outlines a step-by-step approach for image recognition and segmentation:

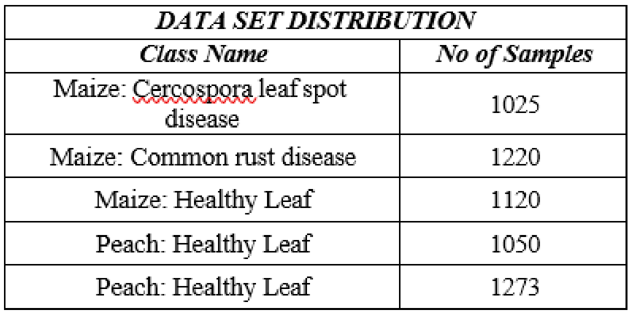
1. **Image Acquisition:** The first step involves capturing an image using a digital camera or similar device.
2. **Preprocessing:** The input image undergoes preprocessing to enhance its quality and remove distortions. Clipping is performed to focus on the region of interest, followed by image smoothing using a smoothing filter. Image enhancement techniques are applied to increase contrast.
3. **Green Pixel Masking:** In this step, mostly green-colored pixels are masked. A threshold value is computed for these pixels. If the pixel intensity of the green component is below the threshold, the red, green, and blue components of that pixel are set to zero.
4. **Removal of Masked Cells:** Masked cells within the infected clusters and their boundaries are removed.
5. **Disease Classification:** The processed image is used as input to a trained Artificial Neural Network (ANN) model, which determines the disease by analyzing probabilities for each class produced by the model.
6. **Fertilizer Recommendations:** If the leaf is infected, a database is queried to provide a list of fertilizer recommendations for treating the disease.

A CNN convolves learned features with input data and uses 2D convolutional layers, making it ideal for processing 2D images. Compared to other image classification algorithms, CNNs require minimal pre-processing, as they can learn filters that otherwise need to be handcrafted in traditional algorithms.

CNNs have a wide range of applications, including image and video recognition, image classification, recommender systems, natural language processing, and medical image analysis. These networks consist of an input layer, an output layer, and hidden layers. The hidden layers typically include convolutional layers, ReLU layers, pooling layers, and fully connected layers. Convolutional layers apply a convolution operation to the input and pass the processed information to the next layer. This methodology demonstrates the practical application of CNNs for the early detection of plant diseases, offering a robust and efficient solution to agricultural challenges.

### 3.1. Dataset Collection and Preparation

The collected dataset consists of RGB images, with an approximately equal distribution of infected and healthy leaf samples. To optimize storage and processing, the images were compressed to reduce the dataset size and categorized into five classes. The dataset distribution is presented in **Table 3.1** below:



| ***DATA SET DISTRIBUTION*** | |
|---|---|
| ***Class Name*** | ***No of Samples*** |
| Maize: Cercospora leaf spot disease | 1025 |
| Maize: Common rust disease | 1220 |
| Maize: Healthy Leaf | 1120 |
| Peach: Healthy Leaf | 1050 |
| Peach: Healthy Leaf | 1273 |

After collecting the images, the dataset underwent processing to ensure it could be efficiently used as input for the Convolutional Neural Network (CNN) algorithm. Tools like OpenCV and Matplotlib were used for image processing. The class labels were encoded using a One Hot Encoder to map alphanumeric labels to numeric values, making prediction easier.

The images were shuffled to ensure a good redistribution of data before splitting. The dataset was then divided into a training set and a validation set to prevent overfitting of the machine learning model.

Finally, the images were reshaped into 3-dimensional tensors with a shape of (28, 28, 3), where:

- **Width** = 28 pixels
- **Height** = 28 pixels
- **Channels** = 3 (for RGB)

This reshaping ensures that all images have uniform dimensions, enabling consistency in input to the CNN model.

### 3.2. Implementing the CNN Algorithm

**Table 3.2** ***CNN MODEL METRICS***

| ***CNN MODEL METRICS*** | | |
|---|---|---|
| ***Layer No*** | ***Layer*** | ***Dimension*** |
| 1 | Input | (28 x 28x 32) |
| 2 | Max Pooling | (14 x 14 x 32) |
| 3 | Convolution + ReLu | (14 x 14 x 64) |
| 4 | Max Pooling | (7 x 7 x 64) |
| 5 | Convolution + ReLu | (17 x 7 x 128) |
| 6 | Dense | (128) |
| 7 | Output | (5) |

The convolution step is the first in a CNN model and is used to extract features from the input image. The input image serves as both a feature indicator and a feature map. A filter is applied block by block to the input image using matrix multiplication. This process highlights the important features within the image.

International Journal of Current Research and Techniques (IJCRT)
E-ISSN: 2349-3194 (Online) | P-ISSN: 2348-4446 (Print)
UGC Approved Journal No: 47722 [No: 2616]
DOI: https://doi.org/10.61359/2024050044

The ReLU Layer (Rectified Linear Unit) follows the convolution layer. It introduces non-linearity into the network by applying an activation function to the feature maps. Since images are inherently non-linear, ReLU eliminates negative values from the activation map by setting them to zero, enhancing the network's ability to learn complex patterns.

Max Pooling involves partitioning the input image into non-overlapping areas and outputting the maximum value from each region. This reduces the image size and the number of parameters, simplifying the computational process while preserving the most significant features.

For implementing the CNN model, various packages from TensorFlow and Keras were used. These tools enable efficient development and training of convolutional neural networks.

### 3.3 Saving the Trained Model

Once the machine learning model has been trained, it is beneficial to save the model along with its parameters. Saving the trained model allows it to be reloaded later for classification tasks without the need for retraining, saving both time and system resources. This approach results in faster classifications.

A Keras machine learning model consists of the following components:

1. **Architecture/Configuration**: Specifies the layers in the model and how they are connected.
2. **Weights**: Represents the "state" of the model.
3. **Optimizer**: Defined during the model compilation process.
4. **Losses and Metrics**: Defined during compilation or through the add_loss() or add_metric() functions.

The Keras API facilitates saving these components to disk either together or selectively:

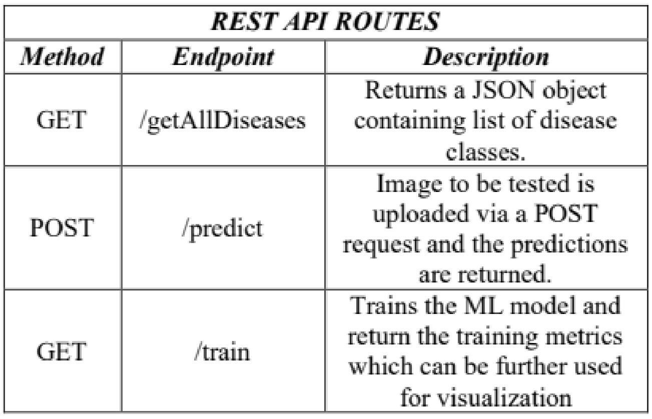
- **Saving everything**: Stored as a single archive in the TensorFlow SavedModel format or the older Keras H5 format.
- **Saving architecture only**: Typically stored as a JSON file.

This flexibility makes it easier to manage models for reuse, ensuring efficient deployment of trained networks.

### 3.4 Implementing the Backend

After saving the trained model, the next step is to load the model and perform classification on an uploaded leaf sample. For implementing the backend, we selected the **Flask** library. Flask was used to create REST APIs, enabling direct access to server methods independently of the user interface. This approach allowed the same APIs to be consumed by various frontend platforms, including an Android app and a web interface.

The following REST APIs were created to classify images and return results as JSON responses:



| ***REST API ROUTES*** | | |
|---|---|---|
| ***Method*** | ***Endpoint*** | ***Description*** |
| GET | /getAllDiseases | Returns a JSON object containing list of disease classes. |
| POST | /predict | Image to be tested is uploaded via a POST request and the predictions are returned. |
| GET | /train | Trains the ML model and return the training metrics which can be further used for visualization |

**Steps Performed During Image Classification**

1. **Uploading the Image:** Users upload an image via the frontend, which sends a POST request to the appropriate API route. The image is passed as content in the request.
2. **Saving the Image:** The server temporarily saves the uploaded image to a designated location. This temporary file is indexed into a variable and flushed after classification.
3. **Loading the Model:** The server loads the pre-trained machine learning model for processing the image.
4. **Image Preprocessing:** The uploaded image is resized to match the input dimensions expected by the model (28 x 28 x 3).
    - The **Matplotlib** library is used to resize the image.
    - The image is converted into a NumPy array using the **OpenCV** library for compatibility with the model.
5. **Classifying the Image:** The processed image is passed through the model for classification. The model returns an array of probabilities for each class. The class label with the highest probability is identified as the final prediction.
6. **Returning the Results:** The server sends the classification results back to the frontend as a JSON response. The response includes:
    - The probability for each class.
    - The class with the highest probability, indicating the final prediction.

**Hosting the Web Application**

Once the Flask web app was developed, it was hosted on an online server to make it accessible from anywhere via a server link. Heroku Web Hosting was chosen for deployment due to its ease of use and suitability for the project requirements.

**Android App Features**

An Android app was developed to complement the web application. The app includes the following features:

1. **Image Upload**
    - A button to upload an image from the phone's camera or gallery for disease detection.
2. **Model Details**
    - An option to view the machine learning model’s details, including model statistics and training metrics.
3. **Tutorial Page**
    - A guide on how to use the app effectively.
4. **Helpful Links**
    - Links redirecting to government-operated agriculture and farmer welfare sites.
5. **Support Feature**
    - A button to directly call the nearest toll-free Kisaan Care Center.

By integrating these features, the system provides an accessible and user-friendly interface for disease detection and agricultural support.

## 4.Results and Conclusion

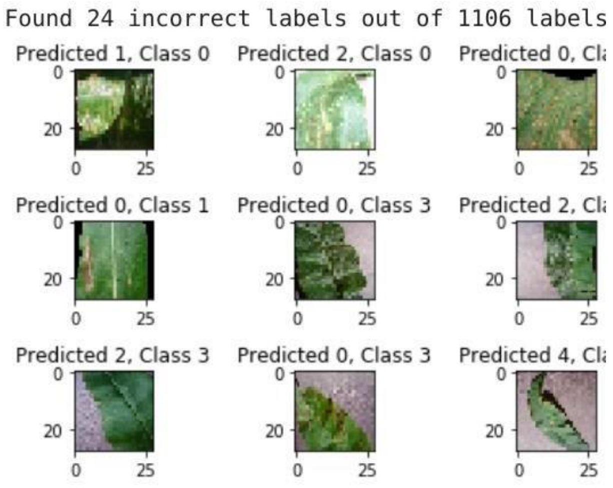

```
Found 24 incorrect labels out of 1106 labels
```

**Figure 4.1. Incorrect Classification of Image Samples**

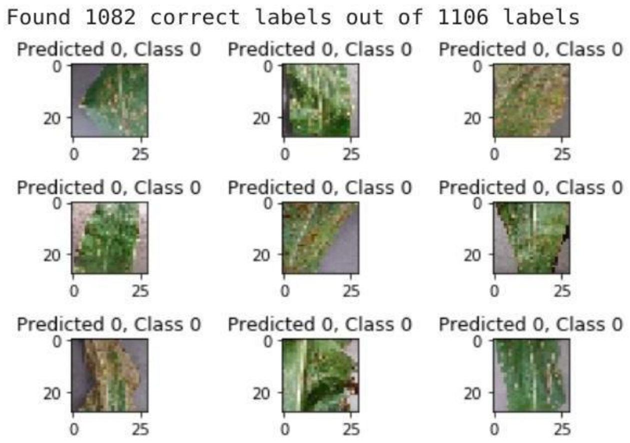
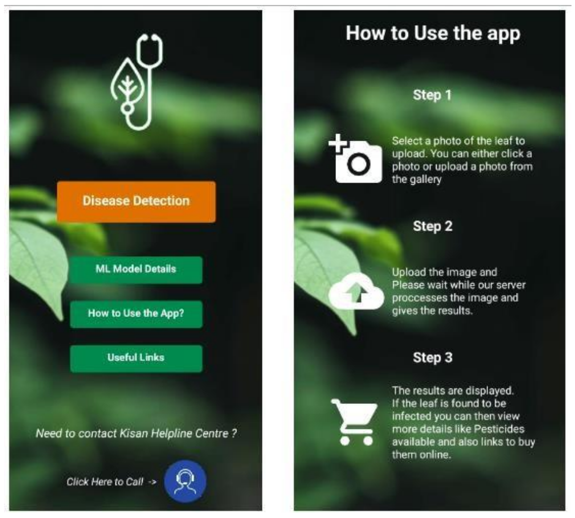

```
Found 1082 correct labels out of 1106 labels
```

**Figure 4.2. Correct Classification of Image Samples**



How to Use the app

Step 1

Select a photo of the leaf to upload. You can either click a photo or upload a photo from the gallery

Step 2

Upload the image and Please wait while our server proccesses the image and gives the results.

Step 3

The results are displayed. If the leaf is found to be infected you can then view more details like Pesticides available and also links to buy them online.

**Figure 4.3. Android App**

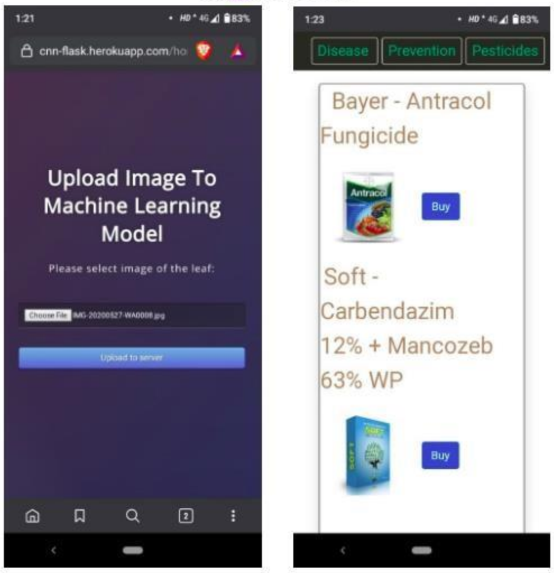

1:21
cnn-flask.herokuapp.com/ho
Upload Image To Machine Learning Model
Please select image of the leaf:
Choose File IMG-20200527-WA0008.jpg
Upload to server
1:23
Disease
Prevention
Pesticides
Bayer - Antracol Fungicide
Buy
Soft - Carbendazim 12% + Mancozeb 63% WP
Buy

**Figure 4.4. Classification and Pesticide Recommendations**

Our automated leaf disease detection solution, based on an artificial neural network, offers a robust and efficient technique for plant leaf disease classification. The primary advantages of our system include its high processing speed and classification accuracy. Its user-friendly interface ensures quick results, eliminating long wait times for diagnosis. Additionally, the system recommends suitable fertilizers and pesticides for treating the identified diseases, along with links to websites where these products can be purchased online. Thus, our solution serves as a comprehensive tool, enabling users to identify plant diseases and take prompt measures to counteract them.

## 5. Future Enhancements

While the current implementation supports the classification of a limited subset of diseases, future developments can focus on improving the model's precision and expanding its scope. Extensive training on diverse datasets—including images from various plants and environmental conditions—would enable the model to classify a broader range of plant diseases accurately. Furthermore, considering the potential use of the system in areas with limited internet connectivity, implementing data compression strategies could significantly reduce the amount of data exchanged with the server. This optimization would result in even faster processing and enhanced usability in remote regions.

## 6. References

[1] Annabel, L. S. P., Annapoorani, T., & Deepalakshmi, P. (2019). Machine learning for plant leaf disease detection and classification – A review. International Conference on Communication and Signal Processing (ICCSP). IEEE.
[2] Singh, V., & Misra, A. K. (2017). Detection of plant leaf diseases using image segmentation and soft computing techniques. Information Processing in Agriculture, 4(1), 41–49.
[3] Muthukannan, K., Latha, P., Pon Selvi, R., & Nisha, P. (2016). Classification of diseased plant leaves using neural network algorithms. Journal of Computer Applications, 10(4).
[4] Mengistu, A. D., Alemayehu, D. M., & Mengistu, S. G. (2016). Ethiopian coffee plant diseases recognition based on imaging and machine learning techniques. International Journal of Computer Applications.
[5] Al-Bashish, D., Braik, M., & Bani-Ahmad, S. (2014). Detection and classification of leaf diseases using K-means-based segmentation and neural-networks-based classification. Information Technology Journal, 10(3), 267–275.
[6] Savita, N. (2014). Detection and classification of plant leaf diseases using image processing techniques: A review. International Journal of Recent Advances in Engineering and Technology.
[7] Dhaygude, S. B., & Kumbhar, N. P. (2013). Agricultural plant leaf disease detection using image processing. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering.
[8] Kulkarni, A. H., & Patil, R. K. A. (2014). Applying image processing techniques to detect plant diseases. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering.