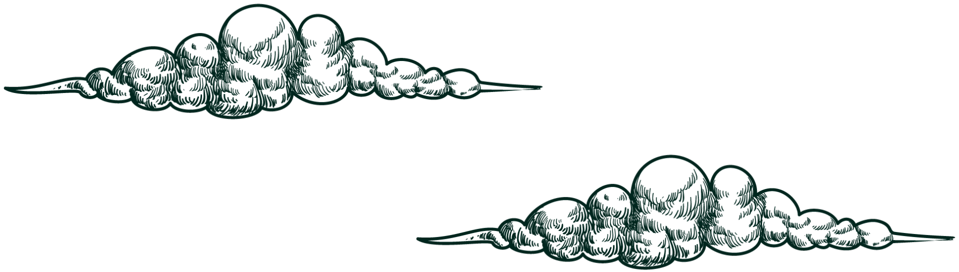


Agro Vision

A Deep learning based plant disease detection model

Table of contents

Problem Statement: Addressing Plant Disease Identification Challenges for Farmers"

Agriculture is a critical sector that sustains human life, but farmers often face significant challenges in maintaining healthy crops. Plant diseases can severely impact crop yields, leading to economic losses and food insecurity. Many farmers lack access to timely and accurate diagnosis tools for plant diseases, which often requires specialized knowledge and equipment.

Solution?: Automated Plant Disease detection system

Our project aims to bridge this gap by creating a **user-friendly web application** that helps farmers identify plant diseases. Farmers can **upload a photo of a plant's leaf** showing potential signs of disease. Using **Convolutional Neural Networks (CNNs)** and advanced **image processing techniques**, the system will:As a Free user, you are allowed to:

- **Upload Image :** Upload an image of the plant's leaf showing potential signs of disease for analysis.
- **Identify the disease** : Provides the name of the disease affecting the plant
- **Offer diagnosis insights** : Give a brief diagnosis and recommended actions to mitigate or treat the disease.

Solution?: Automated Plant Disease detection system

Our project aims to bridge this gap by creating a **user-friendly web application** that helps farmers identify plant diseases. Farmers can **upload a photo of a plant's leaf** showing potential signs of disease. Using **Convolutional Neural Networks (CNNs)** and advanced **image processing techniques**, the system will:As a Free user, you are allowed to:

- **Upload Image :** Upload an image of the plant's leaf showing potential signs of disease for analysis.
- **Identify the disease** : Provides the name of the disease affecting the plant
- **Offer diagnosis insights** : Give a brief diagnosis and recommended actions to mitigate or treat the disease.

Goals:

- **Accessibility:** Make the tool easily usable by farmers with minimal technical knowledge.
- **Efficiency:** Deliver accurate results in a matter of seconds.
- **Scalability:** Support a wide range of plant species and diseases.
- **Awareness:** Help farmers take proactive measures to protect their crops.

Data Collection 📷

Our plant disease detection model was trained on a comprehensive dataset comprising highquality images of both healthy and diseased plant leaves across various crop species.
For example overview of some of the crops:

Potato	Comprises healthy leaves and those aected by Early Blight and Late Blight.
Orange	Contains images of leaves aected by Huanglongbing (Citrus Greening) disease.
Tomato	Contains a wide range of conditions, including healthy leaves and leaves aected by Bacterial Spot, Early Blight, Late Blight, Leaf Mold, Septoria Leaf Spot, Spider Mites (Twospotted Spider Mite), Target Spot, Tomato Yellow Leaf Curl Virus, and Tomato Mosaic Virus.
StrawBerry	Includes images of healthy strawberry leaves and those with Leaf Scorch.
Corn (maize)	Comprises healthy corn leaf images alongside images showing Cercospora Leaf Spot, Common Rust, and Northern Leaf Blight.
Apple	: Contains images of healthy apple leaves as well as leaves aected by diseases such as Apple Scab, Black Rot, and Cedar Apple Rust.

DataSet Preprocessing

- **Image Resizing**:
  Ensures all images are reshaped to a consistent size of **(128, 128)**, enabling seamless input to the CNN model and maintaining uniformity across the dataset.

- **Batching**:
  Divides the dataset into smaller groups (batch size = **32**) for efficient memory usage and faster training by processing multiple images simultaneously.

- **Shuffling**:
  Randomizes the order of images to prevent learning biases caused by dataset organization and ensures better generalization.

- **Automatic Label Inference**:
  Class labels are automatically assigned based on folder names, simplifying the labeling process and ensuring accuracy.

- **Categorical Label Encoding**:
  Labels are transformed into one-hot encoded vectors, making them suitable for multi-class classification tasks.

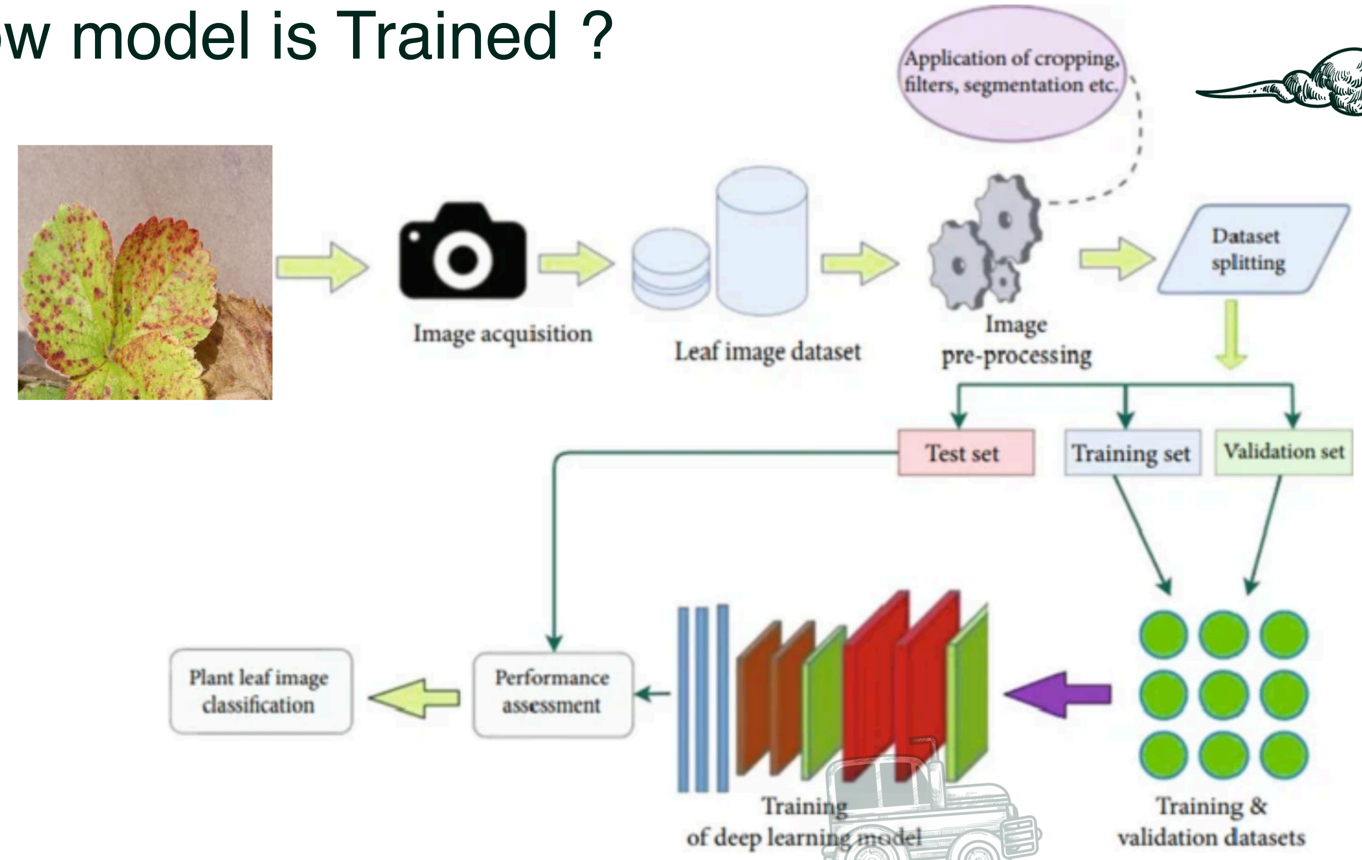
- **RGB Conversion**:
  Ensures all images are loaded in **RGB format**, maintaining consistency in color channels (3 channels: Red, Green, Blue).

- **Interpolation for Resizing**:
  Uses **bilinear interpolation** to resize images smoothly, preserving visual quality and feature integrity.

Model Overview

Our model is a **Convolutional Neural Network (CNN)** designed for **multi-class image classification** with 38 classes. It uses **convolutional layers** to extract features from images, followed by **max pooling** to downsample and **dropout** layers to prevent overfitting. The model includes a **flatten** layer to prepare for fully connected layers and ends with a **softmax output** layer for classification. It is compiled with the **Adam optimizer** and **categorical cross-entropy** loss for efficient training. The model is evaluated based on **accuracy** on both training and validation data. It is designed to classify images efficiently, leveraging best practices like **ReLU activation** and **regularization**. Potential enhancements include **data augmentation** and **transfer learning**.

How model is Trained ?



Model Architecture

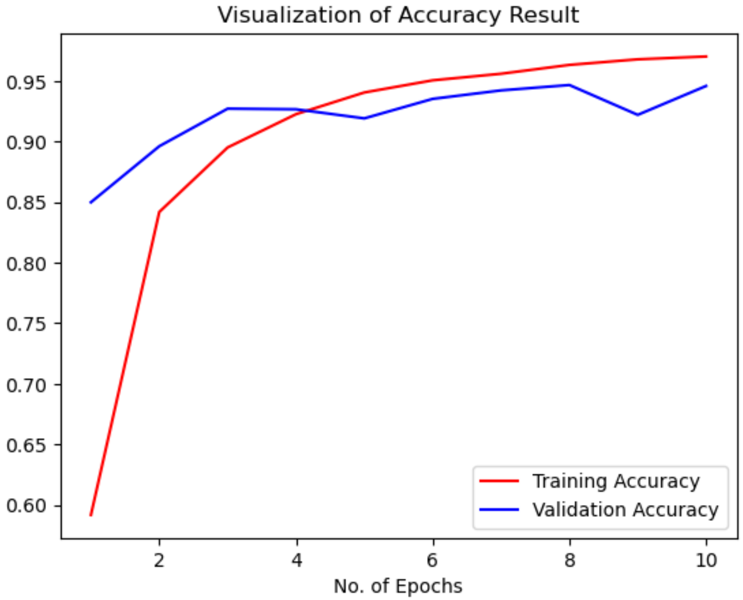
- **Convolutional Layers**: Extract features from images using filters, followed by **ReLU activation** for non-linearity.
- **Max Pooling**: Reduces the spatial dimensions of feature maps while retaining essential information.
- **Dropout**: Prevents overfitting by randomly setting some units to zero during training.
- **Flatten Layer**: Converts 2D feature maps into a 1D vector for input into fully connected layers.
- **Fully Connected (Dense) Layers**: A 1500-unit dense layer followed by a **softmax output layer** with 38 units, one for each class.

Model Training

Optimizer: The model uses the **Adam optimizer** with a learning rate of 0.0001 for efficient weight updates during training.
Batch Size: The model processes **32 images per batch** during training, allowing efficient use of memory and computation.
Loss Function: **Categorical cross-entropy** is used to calculate the loss for multi-class classification, guiding the model to minimize error.
Training Process: The model is trained by iterating over batches of images, adjusting weights to reduce loss improve **accuracy**.

Model Evaluation

- **Evaluation on Training and Validation Data**: After training, the model is evaluated using both the **training dataset** and a **validation dataset** to assess its performance on known and unseen data.
- **Metrics**: The model's **accuracy** is computed, which measures the percentage of correctly classified images. This helps in understanding how well the model is performing.
- **Loss Calculation**: The **loss** is calculated during evaluation using the **categorical cross-entropy** loss function, showing how far the predictions are from the true labels.
- **Overfitting Check**: By comparing accuracy and loss between training and validation sets, you can detect if the model is overfitting (i.e., performing well on training data but poorly on validation data).



Our Team and Work Distribution

Data Collection and Preprocessing

Assigned to: Khushal

- Responsible for collecting and curating a high-quality dataset of plant images with disease labels.
- Preprocess images by resizing, normalizing, and augmenting the dataset through transformations (flipping, rotation, scaling, etc.).
- Split the dataset into training, validation, and testing sets while ensuring class balance.
- Perform data analysis to identify issues like noise or biases in the data.

Our Team and Work Distribution

Model Development and Training

Assigned to: Sai Teja

- Design and implement the CNN model architecture
- Define loss functions, optimization algorithms, and evaluation metrics.
- Train the model using the preprocessed dataset and fine-tune hyperparameters for optimal performance.
- Evaluate the model on validation and test datasets to ensure accuracy and generalization.

Our Team and Work Distribution

Deployment and Diagnosis System

Assigned to: **Srikhar**

- Integrate the trained model into a user-friendly system for client interaction.
- Build a web-based or mobile application that allows users to upload plant images and receive disease predictions.
- Implement the diagnosis component to provide brief descriptions and possible solutions for identified diseases.
- Deploy the model using cloud platforms or local servers and ensure scalability and performance.

Thank You

Does anyone have any questions?

Feel free to ask

dontask@yourmail.com