**PLANT DISEASE DETECTION SYSTEM FOR SUSTAINABLE AGRICULTURE**

WEEK 1 PROJECT:

"Early and accurate detection of plant diseases is a critical challenge in agriculture, especially for small and medium-scale farmers who lack access to expert knowledge. The absence of timely diagnosis leads to reduced crop yield, increased use of harmful pesticides, and long-term soil degradation. This project aims to develop an AI-powered plant disease detection system using image processing and machine learning to enable sustainable farming practices through early intervention, optimized pesticide use, and improved crop management."

**🧩 Breakdown of Sub-Problems**

1. **Lack of Early Detection:** Most farmers rely on visual inspection, which is often inaccurate and delayed, causing irreversible damage to crops.
2. **Dependence on Experts:** Remote and rural areas lack agricultural experts, leading to misdiagnosis or no diagnosis at all.
3. **Overuse of Chemicals:** In the absence of specific diagnosis, farmers use excessive or incorrect pesticides, affecting soil health and food safety.
4. **Limited Technological Adoption:** Existing tools are either too expensive or require complex infrastructure, making them inaccessible to smallholder farmers.
5. **Crop-Specific Knowledge Gaps:** Different plants exhibit unique symptoms of disease that are not always easy to distinguish without expert models or datasets.

**🌱 Sustainable Agriculture Angle**

The system promotes sustainable agriculture by:

* Reducing unnecessary pesticide use.
* Increasing crop yield through timely disease control.
* Enhancing farmer autonomy via mobile-accessible tools.
* Encouraging data-driven decisions in farming.

Here’s a concise and clear explanation of **ANN**, **RNN**, and **CNN** about their structure, use case and limitations..

**1. ANN – Artificial Neural Network**

* The basic form of a neural network, inspired by how human brains process information.
* **Structure:** Consists of input, hidden, and output layers with neurons fully connected.
* **Use Case:** Suitable for simple classification tasks or where data is structured (like numeric crop yield predictions).
* **Limitation:** Does not capture spatial or temporal relationships well, which are important in image or time-series data.

✅ **In Plant Disease Detection:** Can be used, but not ideal alone for analyzing images since it doesn't understand spatial hierarchies (like leaf spots or texture patterns).

**2. RNN – Recurrent Neural Network**

* A type of neural network designed to handle **sequential data** (e.g., time series, speech, text).
* **Feature:** Has a memory loop that feeds past output back into the network, making it good for tracking temporal patterns.
* **Use Case:** Best for tasks like crop disease progression prediction over time, weather-based yield forecasting, etc.
* **Limitation:** Struggles with long-term dependencies and is not commonly used for image classification.

✅ **In Plant Disease Detection:** Useful **only** if your system involves time-based analysis—like detecting disease progression over days/weeks.

**3. CNN – Convolutional Neural Network**

* A specialized neural network designed to process **image data** by automatically extracting features (edges, textures, shapes).
* **Structure:** Uses convolutional layers that apply filters to the image, capturing patterns hierarchically.
* **Use Case:** Dominates in computer vision tasks like image classification, object detection, etc.
* **Strength:** Reduces manual feature extraction and is very efficient in recognizing visual patterns.

✅ **In Plant Disease Detection:** This is the **best model** to identify diseases from leaf images. It detects symptoms like discoloration, spots, lesions, etc.

**PIPELINE CONCEPTS:**

**Data Collection:**

* Gather plant leaf images from:
  + Public datasets (e.g., Plant Village)
  + Field photos using smartphones
* Label each image (e.g., crop name, disease type)
* Use clear, focused images under various lighting

**Data Preprocessing:**

* Resize all images (e.g., 224x224)
* Normalize pixel values
* Use augmentation (rotate, flip, zoom) to improve variety
* Organize in folders: train/, test/, healthy/, diseased

**Data Loading:**

* Use tools like PyTorch or TensorFlow to load images in batches
* Example:

datasets.ImageFolder('dataset/train', transform=...)

**Data Updation:**

* Keep adding new images over time
* Include new diseases/crops.
* Retrain the model regularly to stay updated.

**Zipping and Unzipping method:**

**1. Upload Zip File to Google Drive**

* Go to [Google Drive](https://drive.google.com/)
* Upload your dataset zip file (e.g., plant\_disease\_dataset.zip)

**2. Mount Google Drive in Google Colab**

from google.colab import drive

drive.mount('/content/drive')

**3. Unzip Dataset in Colab**

import zipfile

import os

# Path to the zip file in Google Drive

zip\_path = '/content/drive/MyDrive/plant\_disease\_dataset.zip'

# Extract to a desired folder

extract\_path = '/content/dataset'

# Unzipping

with zipfile.ZipFile(zip\_path, 'r') as zip\_ref:

zip\_ref.extractall(extract\_path)

**4. Check Directory Structure (Optional)**

import os

for root, dirs, files in os.walk(extract\_path):

print(root, len(files))

**5. Use in PyTorch (example)**

from torchvision import datasets, transforms

from torch.utils.data import DataLoader

transform = transforms.Compose([

transforms.Resize((224, 224)),

transforms.ToTensor()

])

train\_dataset = datasets.ImageFolder('/content/dataset/train', transform=transform)

train\_loader = DataLoader(train\_dataset, batch\_size=32, shuffle=True)

**Image Processing and Image Augmentation**

**Image Processing:**

**Common Techniques:**

* **Resizing**: Standardize image size (e.g., 224x224 for CNNs)
* **Normalization**: Scale pixel values (0–1 or mean-std)
* **Colour correction**: Adjust lighting or remove shadows
* **Noise removal**: Filter out blurry or unclear images
* **Segmentation** *(optional)*: Focus only on the leaf, remove background

**Image Augmentation:**

**Common Augmentation Techniques:**

* **Rotation**
* **Flipping (horizontal/vertical)**
* **Zooming**
* **Brightness/contrast adjustment**
* **Shear or warp**
* **Random cropping or shift**

**Basic CNN Model (Keras)**

model = Sequential([

Conv2D(32, (3,3), activation='relu', input\_shape=(224,224,3)),

MaxPooling2D(2,2),

Conv2D(64, (3,3), activation='relu'),

MaxPooling2D(2,2),

Flatten(),

Dense(128, activation='relu'),

Dense(num\_classes, activation='softmax')

])

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

**Use:** For classifying plant diseases from leaf images.

**Model Testing**

* After training, the model is tested using a **separate test dataset** (unseen images).
* This checks how well the model generalizes to new data.

**📊 Model Evaluation Metrics**

1. **Accuracy**: Percentage of correct predictions.
2. **Confusion Matrix**: Shows true vs. predicted classes, helping identify specific class errors.
3. **Precision**: How many predicted diseases are actually correct.
4. **Recall (Sensitivity)**: How well the model identifies all actual disease cases.
5. **F1-Score**: Balance between precision and recall.