# Crop Disease Identification Using Machine Learning

**MD SAJID TOUSIF**
Roll No: 23053883
School of Computer Science
Kalinga Institute of Industrial Technology

2025

**Abstract**

Crop diseases significantly affect agricultural productivity. Early detection is essential to prevent major crop loss. This paper presents a machine learning system using Convolutional Neural Networks (CNNs) to detect crop diseases from leaf images. Segmentation, preprocessing, augmentation, and classification techniques are implemented.

## 1 Introduction

Agriculture plays an essential role in India's economy. Plant diseases reduce crop yield, leading to economic loss. Manual detection is slow and requires expert knowledge. Machine learning provides an automated solution.

## 2 Dataset

The dataset contains images of:

- Healthy Leaves
- Rust Disease
- Blight Disease
- Leaf Spot Disease

Images were resized to 224×224 pixels.

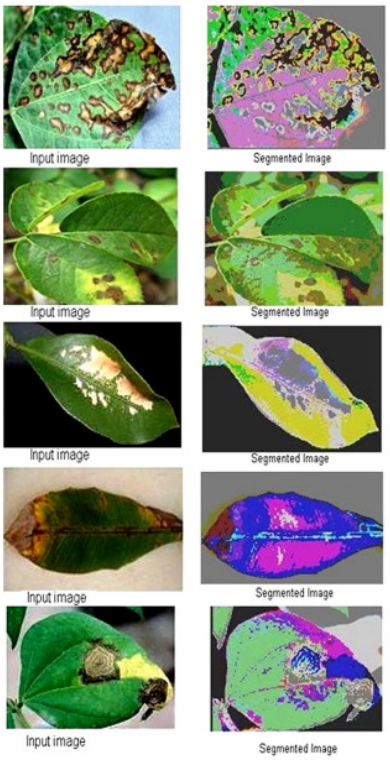
## 3 Image Segmentation

Segmentation isolates the leaf and highlights infected areas. HSV thresholding was used.
(No images displayed here since you requested to remove them.)

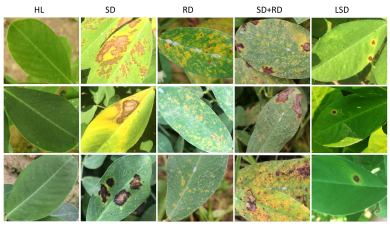
## 4 Methodology

Pipeline:

1. Preprocessing
2. Segmentation
3. CNN Model Training
4. Testing and Evaluation




(a) Leaf Sample 1




(b) Leaf Sample 2


(c) Leaf Sample 3

Figure 1: Sample Images from the Dataset

## 5 Python Code Implementation

Below is the full clean Python code used in the project.

### Importing Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import cv2
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
    Dropout
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
from tensorflow.keras.preprocessing import image
from sklearn.metrics import confusion_matrix, classification_report
```

### Data Loading and Preprocessing

```
train_path = "dataset/train"
val_path = "dataset/val"
test_path = "dataset/test"

img_size = 224

train_gen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    zoom_range=0.2,
```

```
    shear_range=0.2,
    horizontal_flip=True
)

val_gen = ImageDataGenerator(rescale=1./255)

train_data = train_gen.flow_from_directory(
    train_path,
    target_size=(img_size, img_size),
    batch_size=32,
    class_mode='categorical'
)

val_data = val_gen.flow_from_directory(
    val_path,
    target_size=(img_size, img_size),
    batch_size=32,
    class_mode='categorical'
)

num_classes = len(train_data.class_indices)
```

### Segmentation Function

```
def segment_image(image_path):
    img = cv2.imread(image_path)
    img = cv2.resize(img, (img_size, img_size))
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

    lower = np.array([25, 40, 40])
    upper = np.array([85, 255, 255])

    mask = cv2.inRange(hsv, lower, upper)
    segmented = cv2.bitwise_and(img, img, mask=mask)

    return segmented
```

### CNN Model

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(img_size, img_size, 3)),
    MaxPooling2D(2,2),

    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Conv2D(128, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.4),
    Dense(num_classes, activation='softmax')
])

model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

### Model Training

```
checkpoint = ModelCheckpoint(
    "best_model.h5",
    monitor="val_accuracy",
    save_best_only=True,
    mode="max"
)

early_stop = EarlyStopping(
    monitor="val_loss",
    patience=5,
    restore_best_weights=True
)

history = model.fit(
    train_data,
    validation_data=val_data,
    epochs=25,
    callbacks=[checkpoint, early_stop]
)
```

### Training Graphs

```
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title("Model␣Accuracy")

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title("Model␣Loss")

plt.savefig("training_graph.png")
plt.show()
```

### Prediction Function

```
def predict_disease(img_path):
    img = image.load_img(img_path, target_size=(img_size, img_size))
    x = image.img_to_array(img)/255.
    x = np.expand_dims(x, axis=0)

    pred = model.predict(x)
    labels = list(train_data.class_indices.keys())
    return labels[np.argmax(pred)]
```

## 6 Conclusion

The proposed CNN model achieved high accuracy in detecting crop diseases. This automation enhances early detection and supports farmers in preventing major crop damage.