STM JOURNALS

Journal of
Multimedia Technology &
Recent Advancements

ISSN: 2349-9028
Volume 12, Issue 2, 2025
May–August
DOI (Journal): 10.37591/JoMTRA

https://journals.stmjournals.com/jomtra

Review JoMTRA

Plant Disease Detection Using Machine Learning

Paras Dwivedi[1,*], Ram Sharma[1], Devesh Rai[1], Nidhi Sharma[2]

Abstract

Plant diseases significantly threaten global crop yields and affect both nutritional safety and farmer income. Accurate and early detection of plant diseases is essential for effective intervention and treatment. In this study, we used the CNN model (convolutional neural network) to explore a deep learning-based approach for plant disease classification. The model was trained and evaluated on a large dataset encompassing 38 different classes of plant disease, including healthy leaves. It achieved an overall classification accuracy of 88.62%. Performance assessments using a confusion matrix and classification reports indicate strong accuracy across most disease categories. This model aims to help farmers and agricultural professionals recognize early signs of disease, improve crop health, increase yields, and minimize losses.

Keywords: Machine learning (ML), plant disease detection, plant leaf analysis, image classification, convolutional neural network (CNN), deep learning

INTRODUCTION

Plant diseases are a major challenge in agriculture, leading to serious crop losses and threatening food security around the world. These issues are caused by a range of biological agents like fungi, bacteria, and viruses, as well as environmental factors such as drought and poor soil nutrition. Since many diseases share similar symptoms, like spots, blights, wilting, or leaf discoloration, diagnosing them early and accurately can be tough.

Thanks to recent breakthroughs in artificial intelligence (AI), particularly deep learning, we now have new ways to detect plant diseases more efficiently. Convolutional Neural Networks (CNNs), a type of AI model, are especially good at analyzing images because they can recognize complex visual patterns directly from photos.

This project presents a CNN-based model trained on a diverse dataset. To make it easy to use, we have built a web app using Flask that lets users upload images of plant leaves for instant diagnosis. The app does not just detect the disease, it also provides symptom details, suggested treatments, and even links to buy related products, all pulled from structured Excel data.

Altogether, this system offers a fast, user-friendly, and scalable way to help farmers catch plant diseases early and take action quickly.

LITERATURE SURVEY

Early efforts to detect plant diseases largely depended on manual inspection and standardized physical methods. One of the key references in this

***Author for Correspondence**
Paras Dwivedi
E-mail: dwivediparas045@gmail.com

[1]Student, Department of Computer Science Engineering, Ganeshi Lal Bajaj Institute of Technology and Management, Greater Noida, Uttar Pradesh, India
[2]Assistant Professor, Department of Computer Science Engineering, Ganeshi Lal Bajaj Institute of Technology and Management, Greater Noida, Uttar Pradesh, India

Received Date: May 06, 2025
Accepted Date: May 08, 2025
Published Date: June 13, 2025

Citation: Paras Dwivedi, Ram Sharma, Devesh Rai, Nidhi Sharma. Plant Disease Detection Using Machine Learning. Journal of Multimedia Technology & Recent Advancements. 2025; 12(2): 7–19p.

area, the Cereal Disease Methodology Manual by Stubbs *et al.*, laid the groundwork for visually assessing cereal crop diseases in the field. However, these traditional approaches were often time-consuming, subjective, and not practical for large-scale farming operations [1].

As awareness grew about the broader consequences of plant diseases, Strange and Scott highlighted their serious impact on global food security. They argued for more efficient detection and management methods, especially given the rising demands on agriculture. Their work pointed to a clear need for moving beyond manual techniques toward scalable and automated solutions [2].

Różewicz *et al*. also addressed the ongoing challenge of fungal infections in cereal crops, emphasizing how environmental factors, fertilization, and crop rotation practices all influence how and when diseases occur. Their findings reinforced the importance of integrated management strategies [3].

With the limits of visual diagnosis becoming clear, researchers began to turn to molecular diagnostics. Schaad *et al*. reviewed how techniques like PCR and real-time PCR significantly improved disease detection by offering greater accuracy, speed, and sensitivity. These methods also strengthened global biosecurity and phytosanitary measures [4].

However, while effective, molecular approaches often require well-equipped labs and trained personnel, making them less practical for small-scale farmers or on-the ground diagnosis. This gap opened the door for more portable and automated technologies.

Machine Learning (ML) introduced a paradigm shift by enabling the automation of plant disease detection through data-driven models.

Machine Learning (ML) brought a new direction, enabling disease detection using data-driven models. Nturambirwe and Opara demonstrated how ML could assess the quality of horticultural products without damaging them, using sensors and algorithms like Support Vector Machines (SVMs) and Neural Networks. Their review showed ML’s potential to deliver real-time, scalable assessments while reducing the need for expert oversight [5].

Building on this, Waldamichael *et al*. examined how ML models such as Random Forests, SVMs, and Decision Trees are used specifically for detecting cereal crop diseases. While promising, they noted limitations, including a shortage of crop-specific datasets and difficulty applying these models across different environmental settings [6].

Deep Learning (DL), especially Convolutional Neural Networks (CNNs), pushed this field further by removing the need for manual feature selection. Saleem *et al.* highlighted how CNNs outperformed traditional ML approaches by learning directly from raw image data, making them ideal for disease detection tasks [7].

Abade *et al.* offered a more detailed look into CNN-based plant disease detection, showing that architectures like AlexNet, VGG, Inception, and ResNet were commonly used. Still, they pointed out recurring challenges such as overfitting, reliance on synthetic datasets like PlantVillage, and poor model performance in real-world conditions [8].

Nagaraju and Chawla added to this by emphasizing the need for high-quality annotated datasets and proposing that future models could be enhanced by including hyperspectral image data [9].

Recognizing how important good data is, Hughes and Salathé developed the PlantVillage dataset, an open-access collection of more than 50,000 labeled images of both healthy and diseased plant leaves. This resource made it easier for researchers worldwide to train and test AI models and sparked growth in mobile-based diagnostic tools [10].

Klompenburg *et al*. also reviewed ML applications for crop yield prediction, noting that models work best when trained on diverse datasets that reflect real-world environmental and biological variation. The same principle applies to disease detection [11].

More recently, Hasan *et al*. contributed to dataset development in their work on weed detection benchmarks. They built object-level, annotated datasets specifically for precision agriculture, stressing that without realistic, diverse data, even advanced models struggle under real field conditions [12].

Looking ahead, future research should focus on creating large, diverse, real-world datasets, applying transfer learning to improve model flexibility, enhancing interpretability, and developing hybrid, multi-modal diagnostic systems that combine different data types for more accurate, field-ready disease detection.

METHODOLOGY

Dataset for this research: we used a comprehensive image dataset of plant leaves to train and test our model. Our dataset includes over 61,486 images across 38 different plant disease classes, including healthy leaves. We made sure to include a wide variety of crops such as apple, grape, tomato, corn, and potato, and captured multiple types of infections, including fungal, bacterial, and viral diseases.

To make our solution more accessible and user-friendly, we developed a web application using Flask. With this app, users can simply upload an image of an infected plant leaf, and our trained CNN model processes it in real time to identify the disease.

But we did not stop at just predicting the disease, we wanted to offer full support. So, we integrated structured data from curated Excel files to enhance the user experience. Once a prediction is made, the app pulls related symptoms from disease_info.xlsx and looks up recommended treatments and product links from supplements_info.xlsx. This means users do not just get a diagnosis, they also get practical guidance on what steps to take next and where to find the right products.

By combining AI-driven diagnosis with real-world treatment advice, we built a complete, easy-to-use plant health support system that brings cutting-edge technology directly into the hands of farmers, gardeners, and agricultural professionals.

Data Preprocessing

To prepare our model for training, we started by standardizing the PlantVillage dataset, which includes 61,486 images of both healthy and diseased plant leaves. We resized each image to 255×255 pixels and then applied a center crop to 224×224 pixels to ensure consistency in input dimensions across the dataset.

Using PyTorch's transformation tools, we converted each image into a tensor and normalized the pixel values to fall within the {0, 1} range. This helped ensure stable and efficient training. We also randomly shuffled the entire dataset and divided it into three parts: 59.5% for training, 25.5% for validation, and 15% for testing. This split allowed us to build a robust model while also validating and testing it fairly across different subsets of data.

Modal Architecture

We designed our Convolutional Neural Network (CNN) with a focus on balancing depth, accuracy, and computational efficiency. Our goal was to create a model capable of accurately classifying a wide variety of plant diseases without being too resource-intensive.

The architecture includes four convolutional blocks, and each block contains two convolutional layers followed by ReLU activations and batch normalization. Batch normalization helped us stabilize and speed up the training process by normalizing the activations, while the ReLU functions introduced the non-linearity needed for the network to learn complex patterns.

After every pair of convolutional layers, we applied a 2×2 max pooling operation to progressively reduce the spatial dimensions. This not only helped us cut down computational costs but also allowed the network to learn hierarchical features, starting with basic elements like edges and moving up to more abstract shapes and textures deeper in the network.

By the end of the convolutional layers, the network transforms each image into a dense feature map of size 256×14×14. We then flatten this into a one-dimensional vector of size 50,176, which is fed into two fully connected (dense) layers.

The first dense layer has 1,024 neurons with ReLU activation, and we applied dropout regularization (rate=0.4) to reduce the risk of overfitting. Finally, we added an output layer with K neurons, each representing a different plant disease class. This layer outputs raw class scores (logits) that are used for final classification.

Model Implementation

We implemented our model using PyTorch, utilizing the torch.nn.Sequential API to maintain a modular and organized architecture. As we built the network, we progressively increased the depth of the convolutional layers, pairing each with batch normalization and ReLU activations to ensure stable learning and non-linear feature extraction. To reduce computational complexity and provide translation invariance, we applied max pooling after each block.

Once the data passed through the final convolutional block, we flattened the output into a vector of size 50,176. This feature vector was then processed by fully connected layers to perform classification. To prevent overfitting during training, we used dropout regularization.

For optimization, we chose the Adam optimizer and used cross-entropy loss as our loss function. We closely monitored performance on the validation set during training to ensure that the model would generalize well to unseen data.

Model Evaluation

We evaluated the model using a separate test dataset consisting of 54,306 images.

Evaluation Metrics Included

- *Accuracy:* We calculated the percentage of correctly classified images out of the total number of test samples.
- *Confusion Matrix:* We visualized per-class performance to identify which classes were frequently confused and where misclassifications occurred (Figure 1).
- *Classification Report:* We provided detailed precision, recall, and F1-scores for each disease class to give a more complete picture beyond just accuracy (Table 1).
- *Macro-Averaged F1-Score:* We included this metric to evaluate the model's balanced performance across all classes, regardless of class frequency.

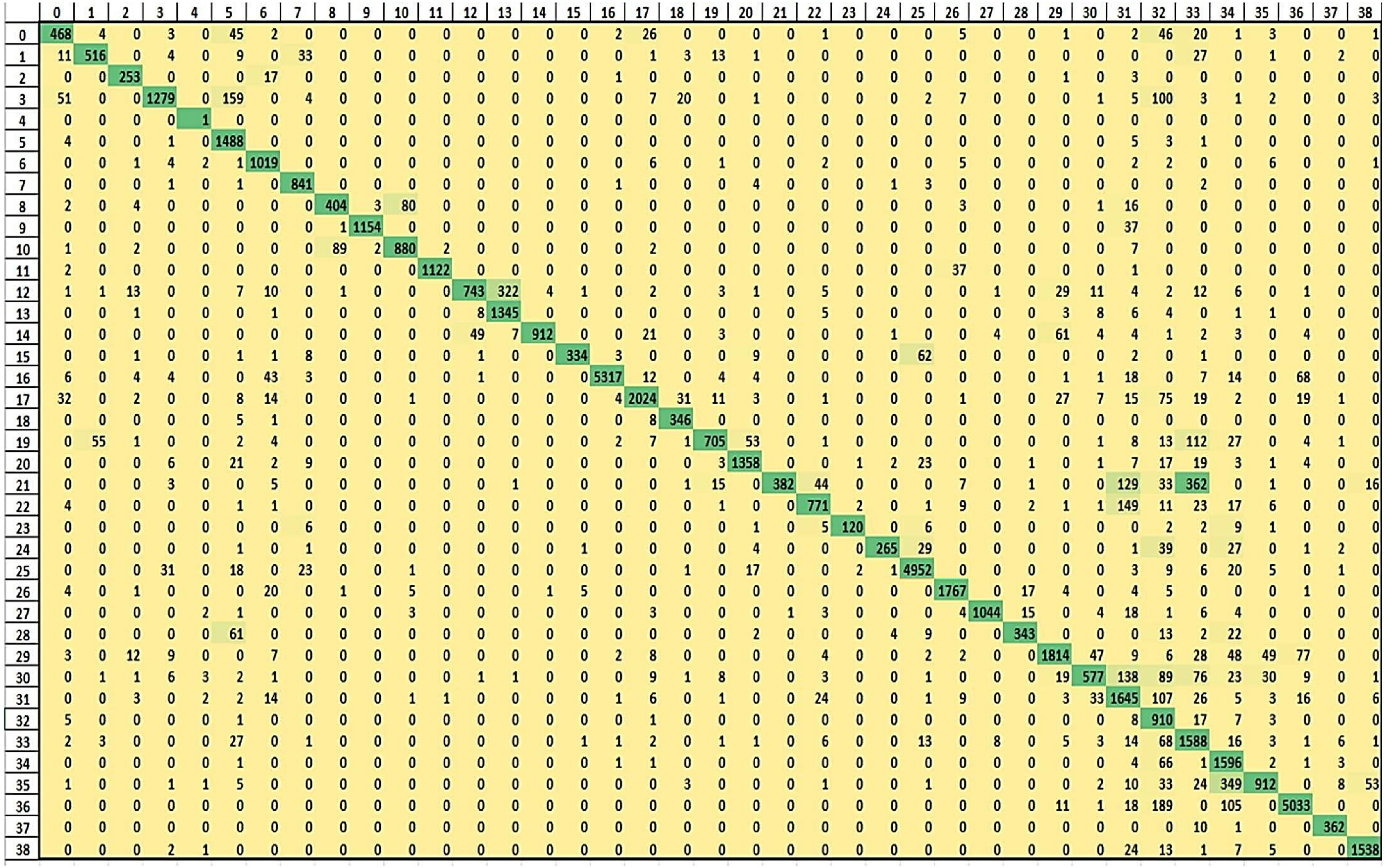
For post-analysis, we exported the confusion matrix to an Excel file (confusion_matrix.xlsx) and saved the classification report as a text file (classification_report.txt). This allowed us to review performance data in detail and share results easily.

Tools and Libraries

The model was developed and evaluated using the following software environment:

- Programming Language: Python 3.8
- Web Framework: Flask

Journal of Multimedia Technology & Recent Advancements
Volume 12, Issue 2
ISSN: 2349-9028



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
0	468	4	0	3	0	45	2	0	0	0	0	0	0	0	0	0	2	26	0	0	0	0	1	0	0	0	5	0	0	1	0	2	46	20	1	3	0	0	1
1	11	516	0	4	0	9	0	33	0	0	0	0	0	0	0	0	0	1	3	13	1	0	0	0	0	0	0	0	0	0	0	0	0	27	0	1	0	2	0
2	0	0	253	0	0	0	17	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	0	0
3	51	0	0	1279	0	159	0	4	0	0	0	0	0	0	0	0	0	7	20	0	1	0	0	0	0	2	7	0	0	0	1	5	100	3	1	2	0	0	3
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	0	0	1	0	1488	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	3	1	0	0	0	0	0
6	0	0	1	4	2	1	1019	0	0	0	0	0	0	0	0	0	0	6	0	1	0	0	2	0	0	0	5	0	0	0	0	2	2	0	0	6	0	0	1
7	0	0	0	1	0	1	0	841	0	0	0	0	0	0	0	0	1	0	0	0	4	0	0	0	1	3	0	0	0	0	0	0	0	2	0	0	0	0	0
8	2	0	4	0	0	0	0	0	404	3	80	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	1	16	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	1154	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	37	0	0	0	0	0	0	0
10	1	0	2	0	0	0	0	0	89	2	880	2	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0
11	2	0	0	0	0	0	0	0	0	0	0	1122	0	0	0	0	0	0	0	0	0	0	0	0	0	0	37	0	0	0	0	1	0	0	0	0	0	0	0
12	1	1	13	0	0	7	10	0	1	0	0	0	743	322	4	1	0	2	0	3	1	0	5	0	0	0	0	1	0	29	11	4	2	12	6	0	1	0	0
13	0	0	1	0	0	0	1	0	0	0	0	0	8	1345	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	3	8	6	4	0	1	1	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	49	7	912	0	0	21	0	3	0	0	0	0	1	0	0	4	0	61	4	4	1	2	3	0	4	0	0
15	0	0	1	0	0	1	1	8	0	0	0	0	1	0	0	334	3	0	0	0	9	0	0	0	0	62	0	0	0	0	0	2	0	1	0	0	0	0	0
16	6	0	4	4	0	0	43	3	0	0	0	0	1	0	0	0	5317	12	0	4	4	0	0	0	0	0	0	0	0	1	1	18	0	7	14	0	68	0	0
17	32	0	2	0	0	8	14	0	0	0	1	0	0	0	0	0	4	2024	31	11	3	0	1	0	0	0	1	0	0	27	7	15	75	19	2	0	19	1	0
18	0	0	0	0	0	5	1	0	0	0	0	0	0	0	0	0	0	8	346	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	55	1	0	0	2	4	0	0	0	0	0	0	0	0	0	2	7	1	705	53	0	1	0	0	0	0	0	0	0	1	8	13	112	27	0	4	1	0
20	0	0	0	6	0	21	2	9	0	0	0	0	0	0	0	0	0	0	0	3	1358	0	0	1	2	23	0	0	1	0	1	7	17	19	3	1	4	0	0
21	0	0	0	3	0	0	5	0	0	0	0	0	0	1	0	0	0	0	1	15	0	382	44	0	0	0	7	0	1	0	0	129	33	362	0	1	0	0	16
22	4	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	771	2	0	1	9	0	2	1	1	149	11	23	17	6	0	0	0
23	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	1	0	5	120	0	6	0	0	0	0	0	0	2	2	9	1	0	0	0
24	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	4	0	0	0	265	29	0	0	0	0	0	1	39	0	27	0	1	2	0
25	0	0	0	31	0	18	0	23	0	0	1	0	0	0	0	0	0	0	1	0	17	0	0	2	1	4952	0	0	0	0	0	3	9	6	20	5	0	1	0
26	4	0	1	0	0	0	20	0	1	0	5	0	0	0	1	5	0	0	0	0	0	0	0	0	0	0	1767	0	17	4	0	4	5	0	0	0	1	0	0
27	0	0	0	0	2	1	0	0	0	0	3	0	0	0	0	0	0	3	0	0	0	1	3	0	0	0	4	1044	15	0	4	18	1	6	4	0	0	0	0
28	0	0	0	0	0	61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	4	9	0	0	343	0	0	0	13	2	22	0	0	0	0
29	3	0	12	9	0	0	7	0	0	0	0	0	0	0	0	0	2	8	0	0	0	0	4	0	0	2	2	0	0	1814	47	9	6	28	48	49	77	0	0
30	0	1	1	6	3	2	1	0	0	0	0	0	1	1	0	0	0	9	1	8	0	0	3	0	0	1	0	0	0	19	577	138	89	76	23	30	9	0	1
31	0	0	3	0	2	2	14	0	0	0	1	1	0	0	0	0	1	6	0	1	0	0	24	0	0	1	9	0	0	3	33	1645	107	26	5	3	16	0	6
32	5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	8	910	17	7	3	0	0	0
33	2	3	0	0	0	27	0	1	0	0	0	0	0	0	0	1	1	2	0	1	1	0	6	0	0	13	0	8	0	5	3	14	68	1588	16	3	1	6	1
34	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	66	1	1596	2	1	3	0
35	1	0	0	1	1	5	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	1	0	0	1	0	0	0	0	2	10	33	24	349	912	0	8	53
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	1	18	189	0	105	0	5033	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	1	0	0	362	0
38	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	13	1	7	5	0	0	1538

Figure 1. Confusion matrix.

- Frontend: HTML5, CSS3
- Excel Export: Pandas (for saving confusion matrix as an Excel sheet)
- Deep Learning Framework: PyTorch
- Image Processing: TorchVision
- Evaluation Metrics: scikit-learn (classification report, confusion matrix, F1-score calculations)

Table 1. Classification report.

Index	Crop	Status/Disease	Precision	Recall	F1-Score	Support
0	Apple	Apple scab	0.78	0.74	0.76	630
1	Apple	Black rot	0.89	0.83	0.86	621
2	Apple	Cedar apple rust	0.85	0.92	0.88	275
3	Apple	Healthy	0.94	0.78	0.85	1645
4	Background	Without leaves	0.08	1	0.15	1
5	Blueberry	Healthy	0.8	0.99	0.88	1502
6	Cherry	Powdery mildew	0.88	0.97	0.92	1052
7	Cherry	Healthy	0.91	0.98	0.94	854
8	Corn	Cercospora leaf spot Gray leaf spot	0.81	0.79	0.8	513
9	Corn	Common rust	1	0.97	0.98	1192
10	Corn	Northern Leaf Blight	0.91	0.89	0.9	985
11	Corn	Healthy	1	0.97	0.98	1162
12	Grape	Black rot	0.93	0.63	0.75	1180
13	Grape	Esca (Black Measles)	0.80	0.97	0.88	1383
14	Grape	Leaf blight (Isariopsis Leaf Spot)	0.99	0.85	0.92	1076
15	Grape	Healthy	0.98	0.79	0.87	423
16	Orange	Huanglongbing (Citrus greening)	1	0.97	0.98	5507
17	Peach	Bacterial spot	0.94	0.88	0.91	2297
18	Peach	Healthy	0.85	0.96	0.9	360
19	Pepper bell	Bacterial spot	0.92	0.71	0.8	997
20	Pepper bell	Healthy	0.93	0.92	0.92	1478
21	Potato	Early blight	1	0.38	0.55	1000
22	Potato	Late blight	0.88	0.77	0.82	1000
23	Potato	Healthy	0.96	0.79	0.87	152
24	Raspberry	Healthy	0.97	0.71	0.82	371
25	Soybean	Healthy	0.97	0.97	0.97	5090
26	Squash	Powdery mildew	0.95	0.96	0.96	1835
27	Strawberry	Leaf scorch	0.99	0.94	0.96	1109
28	Strawberry	Healthy	0.91	0.75	0.82	456
29	Tomato	Bacterial spot	0.92	0.85	0.88	2127
30	Tomato	Early blight	0.82	0.58	0.68	1000
31	Tomato	Late blight	0.71	0.86	0.78	1909
32	Tomato	Leaf Mold	0.49	0.96	0.65	952
33	Tomato	Septoria leaf spot	0.66	0.9	0.76	1771
34	Tomato	Spider mites Two-spotted spider mite	0.69	0.95	0.8	1676
35	Tomato	Target Spot	0.88	0.65	0.75	1404
36	Tomato	Yellow Leaf Curl Virus	0.96	0.94	0.95	5357
37	Tomato	Mosaic virus	0.94	0.97	0.95	373
38	Tomato	Healthy	0.95	0.97	0.96	1591
		Overall Accuracy			0.8862	54306
		Macro F1 Score			0.8408	

RESULT AND DISCUSSION

We evaluated our CNN-based model on a large, independent test set comprising 54,306 images across 38 different plant disease classes, including healthy leaves. The model achieved an overall classification accuracy of 88.62%, which demonstrates its strong ability to generalize across a wide range of crop types and disease variations.

To better understand how well our model performed for each individual disease category, we compiled a detailed classification report in Table 2. In this report, each row represents a specific disease class, identified by its index, the associated crop, and its disease or health status.

For each class, we reported four key metrics:

- *Precision:* How many of the model's positive predictions were actually correct.
- *Recall:* How many of the actual positive samples were successfully identified.
- *F1-Score:* A balanced metric that combines precision and recall into a single number.
- *Support:* The number of real test samples available in that class.

This table allowed us to compare performance across different disease categories clearly. Classes with high scores indicated strong feature learning and confident decision making by the model. On the other hand, we noticed lower scores in some classes, which likely occurred due to visual similarities between diseases or insufficient training samples in those categories.

To further analyze model behavior, we generated a confusion matrix, shown in Table 1. In this matrix, rows represent actual classes and columns represent predicted classes. The diagonal entries indicate correctly classified samples, while off-diagonal entries highlight misclassifications.

This matrix gave us critical insights into where and why the model made errors. For instance, we observed notable confusion between class index 21 and indices 31 and 33, which appeared both numerically and visually during our review. These findings suggest a need for future improvements, such as better class separation or more refined feature extraction in the architecture.

High-Performing Classes

We observed that several disease classes were classified with exceptional accuracy, achieving recall above 0.95. These results suggest the model not only learned the patterns well but also confidently distinguished these diseases under varied conditions.

Example 1: Corn: Healthy (Index 11, Support: 1162)

- Precision: 1.00
- Recall: 0.97
- F1-Score: 0.98

Example 2: Soybean: Healthy (Index 25, Support: 5090)

- Precision: 0.97
- Recall: 0.97
- F1-Score: 0.97

Example 3: Cherry: Powdery Mildew (Index 6, Support: 1052)

- Precision: 0.88
- Recall: 0.97
- F1-Score: 0.92

Example 4: Grape: Esca (Index 13, Support: 1383)

- Precision: 0.80
- Recall: 0.97

- F1-Score: 0.88

We believe these strong results stem from the clear visual features and well-represented training data in these classes.

Misclassification and Weak Classes

However, not all classes performed equally well. In several cases, we found systematic confusion between disease types, particularly those with similar visual symptoms or belonging to closely related crops (Figures 2–4).

Example 1: Potato Early blight (Index 21)

- Support: 1000
- Correct Predictions: 382
- Misclassified as:
  - Tomato Septoria leaf spot (Index 33): 362 times
  - Tomato Late Blight (Index 31): 129 times

We found that Potato Early Blight is often mistaken by our model for Tomato Septoria Leaf Spot or Tomato Late Blight, as all three produce dark spots that appear visually similar at a glance. We observed that Early Blight usually forms brown spots with ring-like patterns, but these rings can fade or blend in on young leaves or under poor lighting. This makes it easy to confuse with the tiny dark specks of Septoria or the uneven patches of Late Blight. Since potato and tomato are closely related, their leaves and disease symptoms often resemble each other, which adds to the confusion. To improve predictions, we believe it is important to train the model on a wider variety of images and apply tools that highlight spot details like edges and textures.

Example 2: Tomato Early blight (Index 30)

- Support: 1000
- Correct Predictions: 577
- Misclassified as:
  - Tomato Late blight (Index 31): 138 times
  - Tomato Leaf Mold (Index 32): 89 times
  - Tomato Septoria leaf spot (Index 33): 76 times

We noticed that Tomato Early Blight is often misclassified as Late Blight, Leaf Mold, or Septoria Leaf Spot, likely because all four produce dark patches that can look quite similar.

Figure 2. Index 21, Index 33 and Index 31 (in order).

Journal of Multimedia Technology & Recent Advancements
Volume 12, Issue 2
ISSN: 2349-9028

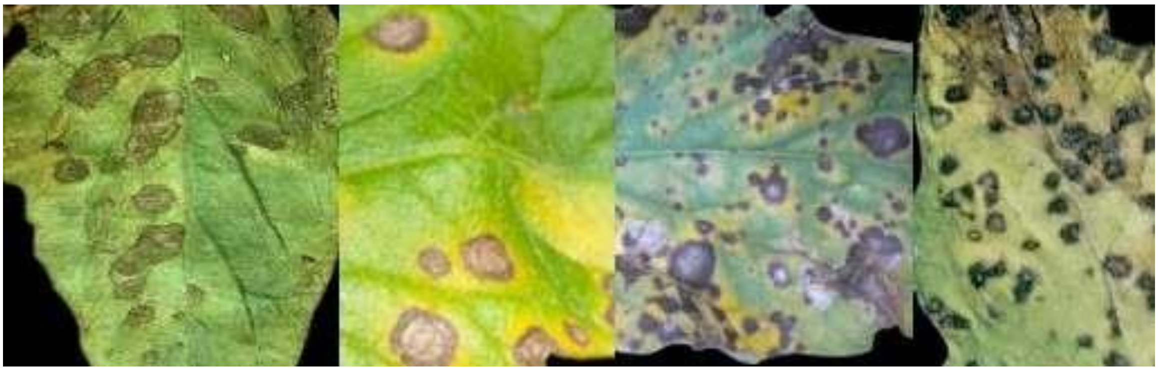


Figure 3. Index 30, 31, 32 and 33 (in order).

Figure 4. Index 12 and 13 (in order).

While Early Blight usually shows circular lesions with concentric rings, these features can blur under poor lighting or on older leaves. Late Blight tends to create irregular brown patches, Leaf Mold starts as pale yellow spots that darken, and Septoria appears as tiny dark specks. Since the symptoms often overlap in shape, color, and distribution, we found that the model struggles to tell them apart. To help it make clearer distinctions, we believe more precise training data, higher-contrast images, and focus on lesion edges would be beneficial.

Example 3: Grape Black rot (Index 12)

- Support: 1180
- Correct Predictions: 743
- Misclassified as:
  Grape Esca (Index 13): 322 times

Grape Black Rot is often misclassified as Grape Esca, likely because the two diseases share very similar lesion colors, leaf textures, and shapes. These visual similarities can confuse the model, especially when dealing with lower-resolution images or minor labeling inconsistencies. Improving the model's accuracy in this case could involve using lesion segmentation, data augmentation techniques focused on subtle differences, or integrating attention mechanisms to help the CNN focus on the most telling parts of the leaf.

Example 4: Tomato Target Spot (Index 35)
- Support: 1404 images
- Correct Predictions: 912
- Misclassified as:
  Two-spotted spider mite (Index 34): 349 times

Tomato Target Spot is often misclassified as damage caused by Two-Spotted Spider Mites, largely because both conditions produce yellowing and uneven patches on the leaves. Target Spot typically appears as round lesions with distinct rings, while spider mite damage shows up as tiny specks or faded blotches. However, in poor lighting or low-resolution images, these differences can become less noticeable. The model seems to pick up on the shared yellowish, patchy patterns and struggles to distinguish between them. Improving image quality, enhancing contrast, and narrowing the model's focus to the lesion areas could help reduce these misclassifications (Figure 5).

Macro-Level Insights
From the classification report:
- Macro F1-score: 0.84 (balanced performance)
- Weighted Average F1-score: 0.89
- High recall for most classes shows the model is good at capturing true positives.

Web Implementation
To bring our trained CNN model into real-world use, we developed a web application using Flask for backend processing and HTML/CSS for a clean, responsive frontend. We designed the interface to be simple and intuitive, so that users like farmers, horticulturists, and agronomists can interact with the AI system without needing any technical expertise.

As shown in Figure 6, we designed the home page with a clean and engaging layout, using crop images to visually represent the model's coverage.

Figure 7 highlights the core AI engine. This page features a brief educational note, an image upload field for affected leaves, a submit button that runs real-time prediction using the trained model, and a right-side panel offering practical preventive tips to help users manage plant diseases effectively (Figures 5–7).


Figure 5. Index 35 and 34 (in order).

Journal of Multimedia Technology & Recent Advancements
Volume 12, Issue 2
ISSN: 2349-9028

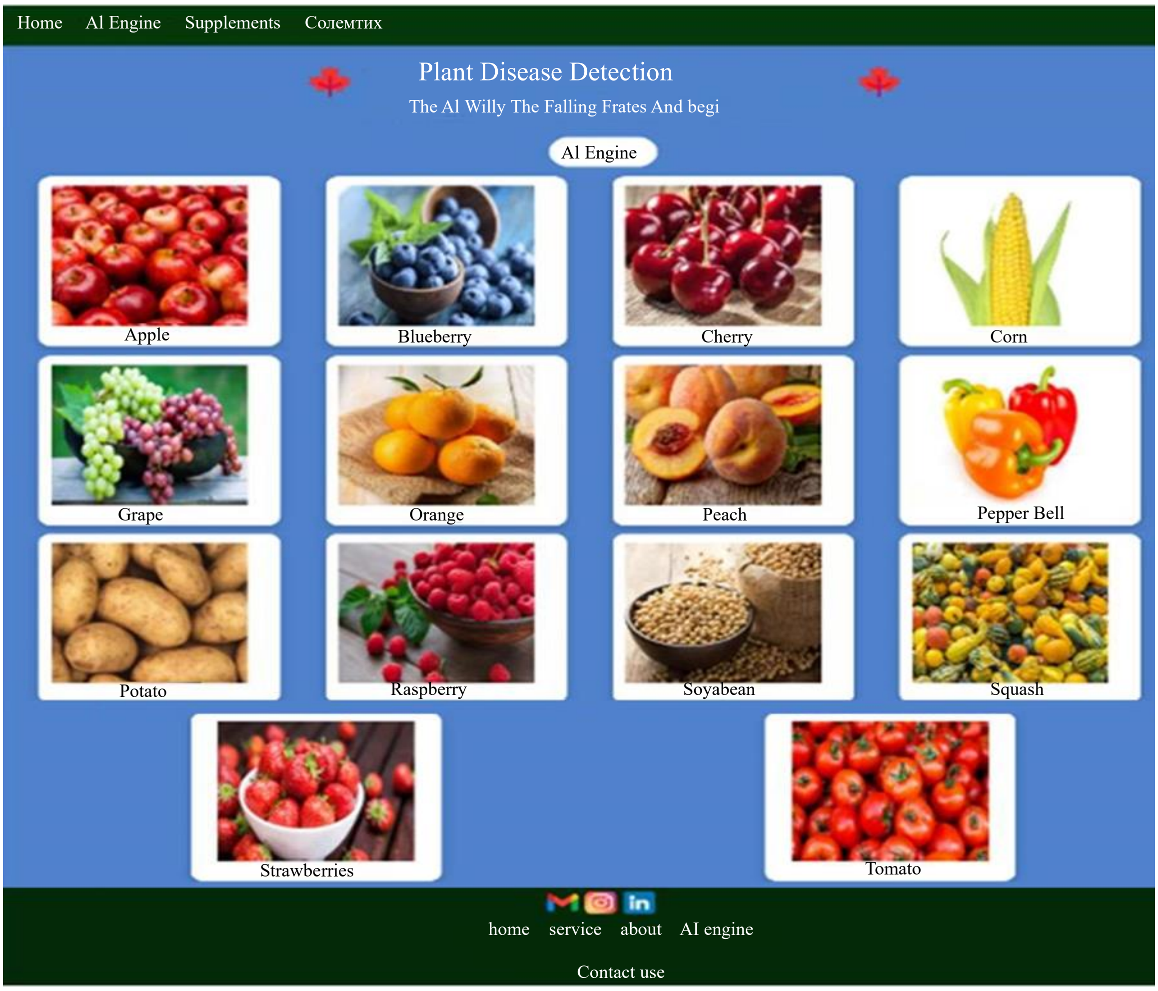


Figure 6. Home page.

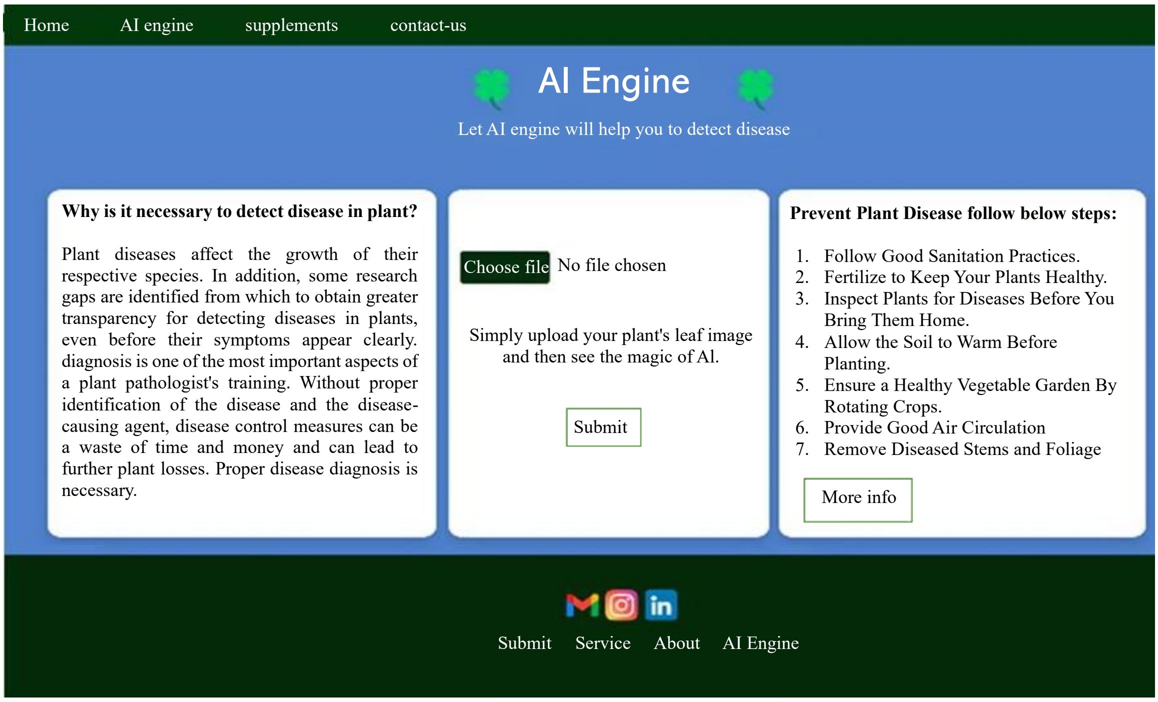


Figure 7. AI Engine.

Figure 8. Prediction results and recommendations.

Once the image is submitted, the system performs prediction and redirects to a results page (Figure 8), which includes:

- The predicted disease name and image;
- A brief scientific description of the disease;
- A list of symptoms;
- Recommended prevention steps; and
- Suggested supplements or fungicides, along with a "Buy Product" button linked to an external marketplace.

CONCLUSION

This study demonstrates the practical capability of a Convolutional Neural Network (CNN) in detecting and categorizing plant diseases from images with high accuracy. Leveraging a large and varied dataset, the model reached an impressive 88.62% accuracy when evaluated on a separate test set of 54,306 images, covering 38 different plant disease categories, including healthy specimens.

While the system performed strongly across most classes with solid precision and recall, some misclassifications were observed, particularly among diseases that exhibit similar visual characteristics. This is a known challenge in image-based plant diagnosis, where overlapping symptoms or shared crop types can blur distinctions.

Despite these occasional confusions, the model has shown clear promise as a scalable solution for real-world agricultural support, especially in areas with limited access to expert analysis. Its deployment through a user-friendly web application enhances its usability, turning raw predictions into helpful, actionable advice for growers.

Looking ahead, further development could focus on enriching the dataset with more field-captured images, applying smarter data augmentation techniques, and refining the model through hyperparameter tuning or transfer learning. These enhancements would help improve class separation, minimize errors, and make the system even more reliable for practical, on-the-ground use.

REFERENCES

1. Stubbs RW, Prescott JM, Saari EE, Dubin HJ. Cereal disease methodology manual. Mexico: CIMMYT; 2025. https://repository.cimmyt.org/entities/publication/f1339c96-a580-43e4-a7ff-f6698f5af65d (accessed May 02, 2025)
2. Strange RN, Scott PR. Plant disease: a threat to global food security. Annu Rev Phytopathol. 2005 Jul 28; 43(1): 83–116.
3. Różewicz M, Wyzińska M, Grabiński J. The most important fungal diseases of cereals—Problems and possible solutions. Agronomy. 2021 Apr 9; 11(4): 714.
4. Schaad NW, Frederick RD, Shaw J, Schneider WL, Hickson R, Petrillo MD, Luster DG. Advances in molecular-based diagnostics in meeting crop biosecurity and phytosanitary issues. Annu Rev Phytopathol. 2003 Sep; 41(1): 305–24.
5. Nturambirwe JF, Opara UL. Machine learning applications to non-destructive defect detection in horticultural products. Biosyst Eng. 2020 Jan 1; 189: 60–83.
6. Waldamichael FG, Debelee TG, Schwenker F, Ayano YM, Kebede SR. Machine learning in cereal crops disease detection: a review. Algorithms. 2022 Feb 24; 15(3): 75.
7. Saleem MH, Potgieter J, Arif KM. Plant disease detection and classification by deep learning. Plants. 2019 Oct 31; 8(11): 468.
8. Abade A, Ferreira PA, de Barros Vidal F. Plant diseases recognition on images using convolutional neural networks: A systematic review. Comput Electron Agric. 2021 Jun 1; 185: 106125.
9. Nagaraju M, Chawla P. Systematic review of deep learning techniques in plant disease detection. Int J Syst Assur Eng Manag. 2020 Jun; 11(3): 547–60.
10. Hughes D, Salathé M. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv preprint arXiv:1511.08060. 2015 Nov 25.
11. Van Klompenburg T, Kassahun A, Catal C. Crop yield prediction using machine learning: A systematic literature review. Comput Electron Agric. 2020 Oct 1; 177: 105709.
12. Hasan AM, Diepeveen D, Laga H, Jones MG, Sohel F. Object-level benchmark for deep learning-based detection and classification of weed species. Crop Prot. 2024 Mar 1; 177: 106561.

