ISSN 2277-2685
IJESR/June. 2025/ Vol-15/Issue-3s/322-329
Panyam Jyoni *et. al.,* / International Journal of Engineering & Science Research

Crop Disease Detection System

P Mounika, Panyam Jyoni, T Karunya
[1]Assistant Professor, Department Of Cse, Bhoj Reddy Engineering College For Women, India.
[2,3]B. Tech Students, Department Of Cse, Bhoj Reddy Engineering College For Women, India.

ABSTRACT
The Crop Disease Detection System is an innovative approach to identifying and diagnosing plant diseases using Convolutional Neural Networks (CNNs). As crop diseases are a significant threat to global food security, timely detection is essential to prevent yield loss. This system leverages deep learning techniques, particularly CNNs, to analyze images of crops and classify them into various disease categories.
The system is trained on a large dataset of labeled crop images, where the CNN is used to automatically extract relevant features such as texture, color, and shape. The model then classifies the images into healthy or diseased crops, identifying specific diseases like fungal infections, bacterial blights, and viral diseases. By utilizing transfer learning and pre-trained models, the system achieves high accuracy even with limited datasets.
The proposed system offers several advantages, including fast and accurate disease detection, cost-effectiveness, and ease of implementation for farmers in the field. It can be deployed on mobile devices or integrated into precision agriculture systems, providing real-time monitoring and early warnings for farmers to take preventive measures. This system aims to reduce the reliance on manual inspections, lower pesticide usage, and ultimately improve crop productivity and sustainability.

1. INTRODUCTION

The Crop Disease Detection System utilizes deep learning, specifically CNNs, to analyze leaf images and detect diseases and pests in crops.Attention mechanisms are integrated to improve the model's focus on critical features, enhancing detection accuracy and minimizing false positives.This system offers farmers real-time, actionable insights, enabling early intervention to improve crop yield and reduce losses.
This project introduces a "Crop Disease Detection System" leveraging the power of Convolutional Neural Networks (CNNs). CNNs are a class of deep learning models specifically designed for image recognition tasks, making them highly suitable for analyzing visual symptoms of plant diseases. By training a CNN model on a vast dataset of healthy and diseased plant images, our system aims to accurately identify various crop diseases at an early stage, providing farmers with actionable insights to protect their crops and maximize their yields. This automated approach promises to be more efficient, accurate, and scalable than traditional methods, ultimately contributing to more sustainable and productive agriculture.
The system enables early identification of crop diseases, pests, and nutrient deficiencies, allowing farmers to take preventive actions before significant damage occurs, leading to better crop health and higher yields.
The AI-driven model can be adapted to monitor various crops and agricultural settings, making it scalable for use in different climates, farming techniques, and regions globally.

Existing System

Existing systems for crop disease detection monitoring rely heavily on manual inspection and traditional methods such as visual observations, which are time-consuming, error-prone, and limited in scale. Farmers must inspect large areas physically, which can lead to delayed detection of issues.

Proposed System

The proposed system utilizes an enhanced Convolutional Neural Network (CNN) integrated with attention mechanisms to accurately analyze leaf images and detect early signs of diseases, pests, and nutrient deficiencies. The system provides real-time analysis of crop health through continuous image data collection, offering instant feedback and alerts to farmers. This allows for timely intervention and targeted treatment, minimizing crop damage and optimizing yield.

2. REQUIREMENT ANALYSIS

Functional Requirements

The Crop Disease Detection System is deep learning application developed to recognize the disease of a crop by taking the leaf image as a input.

Modules:

- **User Module:**

User is responsible for uploading the leaf image.

- Upload a leaf image.
- View the result of crop condition

Non-Functional Requirements

Non-functional requirements specify the system's performance, usability, scalability, and other quality attributes. These are derived from the project's goals of improving trust, performance, and user satisfaction in mobile cloud computing.

1. Performance:

The system should process a single leaf image and provide results within 2 seconds for high-resolution images..

2. Scalability:

It should handle up to 10,000 images daily with minimal performance degradation.

3. Usability:

The interface should be user-friendly, multilingual, and accessible across both desktop and mobile devices.

4. Accuracy:

The system must achieve at least 95% classification accuracy for disease detection

3-DESIGN

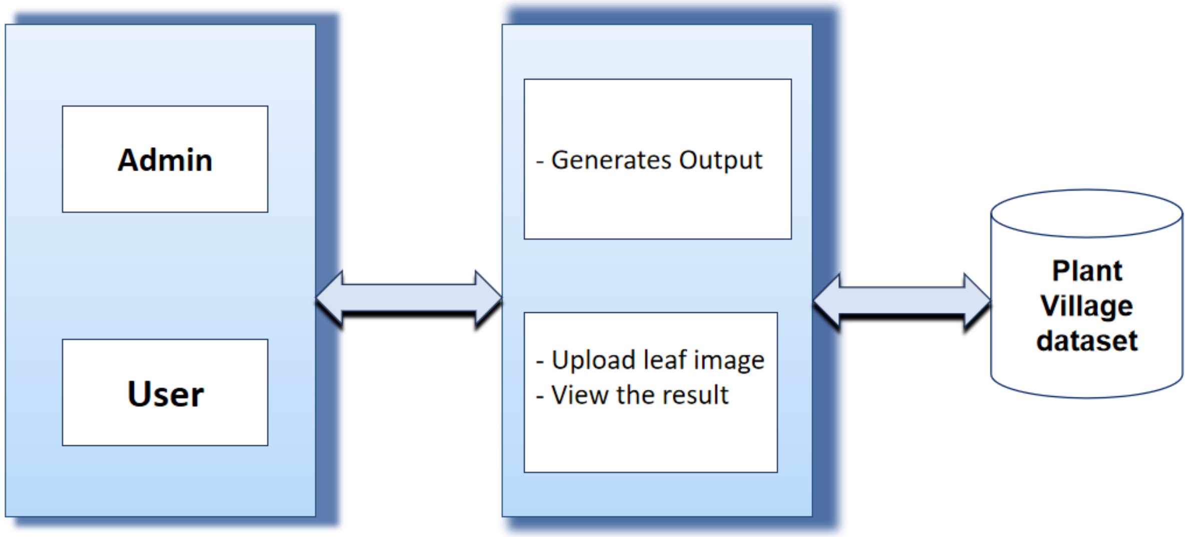
Architecture

Project architecture represents number of components we are using as a part of our project and the flow of request processing i.e. what components in processing the request and in which order. An architecture description is a formal description and representation of a system organized in a way that supports reasoning about the structure of the system. Architecture. is of two types. They are

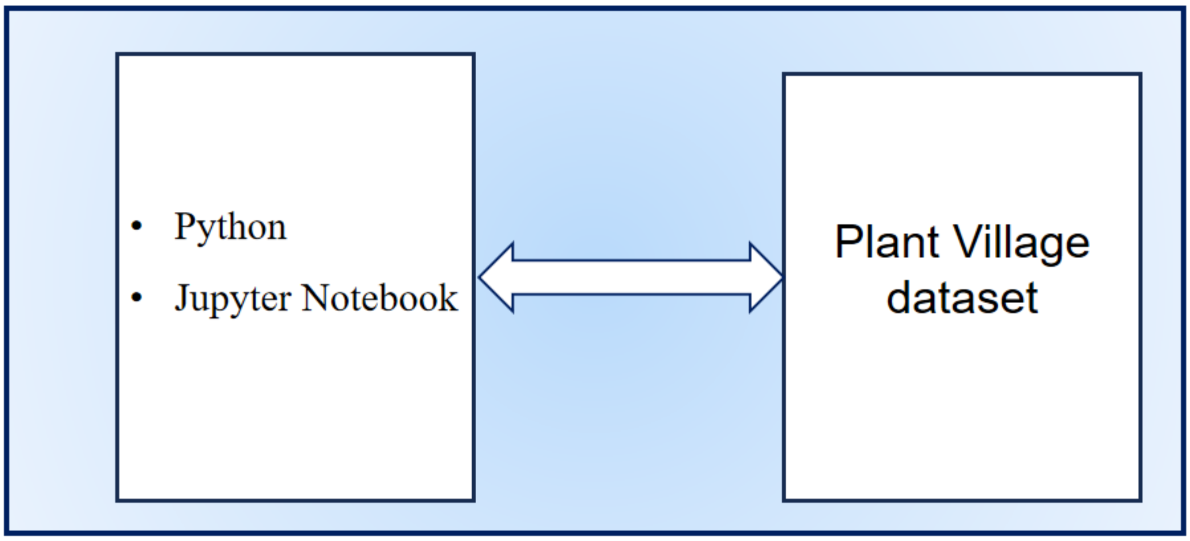
(1) Software Architecture

(2) Technical Architecture

System Architecture



Technical Architecture:



4-IMPLEMENTATION

Python

Python is a programming language that is interpreted, object-oriented, and considered to be high-level too. Python is one of the easiest yet most useful programming languages which is widely used in the software industry. Due to its easiest syntax, it is recommended for beginners who are new to the software engineering field. Its demand is growing at a very rapid pace due to its vast use cases in Modern Technological fields like Data Science, Machine learning, and Automation Tasks. For many years now, it has been ranked among the top Programming languages. Today Python is used in all kinds of development from game development, basic programming, and scripting to large and complex software development. It has a large community support and is rich in the library, having all kinds of frameworks for backend, frontend and you name it python has it all Due to its beginner-friendly syntax, it became a clear choice for beginners to start their programming journey. The major focus behind creating it is making it easier for developers to read and understand, also reducing the lines of code. It has spread its demand in various fields which includes machine learning, artificial intelligence, data analysis, web development.

Deep Learning

In the fast-evolving era of artificial intelligence, Deep Learning stands as a cornerstone technology, revolutionizing how machines understand, learn, and interact with complex data. At its essence, Deep Learning AI mimics the intricate neural networks of the human brain, enabling computers to autonomously discover patterns and make decisions from vast amounts of unstructured data. This transformative field has propelled breakthroughs across various domains, from computer vision and natural language processing to healthcare diagnostics and autonomous driving.

The definition of Deep learning is that it is the branch of machine learning that is based on artificial neural network architecture. An artificial neural network or ANN uses layers of interconnected nodes called neurons that work together to process and learn from the input data.

In a fully connected Deep neural network, there is an input layer and one or more hidden layers connected one after the other. Each neuron receives input from the previous layer neurons or the input layer. The output of one neuron becomes the input to other neurons in the next layer of the network, and this process continues until the final layer produces the output of the network. The layers of the neural network transform the input data through a series of nonlinear transformations, allowing the network to learn complex representations of the input data.

Convolutional Neural Network(CNN):

A Convolutional Neural Network (CNN) is a specialized type of deep learning model primarily used for processing images. CNNs are particularly effective for image classification, object detection, and other computer vision tasks.

1. Convolution Layer:

The heart of a CNN is its convolution operation. This involves a filter (small matrix) sliding over the input image and performing element-wise multiplication followed by summation.

Each filter extracts a specific feature (e.g., edges, patterns) from the image.

2. ReLU Activation Function:

After convolution, the ReLU (Rectified Linear Unit) is applied to introduce non-linearity.

It replaces all negative values in the feature maps with zero, ensuring the network learns complex patterns.

3. Pooling Layer:

Used to reduce the dimensionality of feature while preserving important features.

Common pooling methods:

Max Pooling: Takes the maximum value in a small region.

Average Pooling: Computes the average value in a small region.

Pooling makes the model computationally efficient and helps reduce overfitting.

4. Fully Connected Layer:

- After feature extraction through convolution and pooling, fully connected layers are used for classification.
- They connect every neuron from the previous layer to all neurons in the current layer, acting as a standard neural network.

5. Output Layer:

Produces the final prediction, often using activation functions like:

SoftMax (for multi-class classification).

5. TESTING

Overview

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

Some of the reasons why software testing becomes very significant and integral part in the field of information technology are as follows.

1.Cost effectiveness
2.Customer Satisfaction
3.Security
4.Maintainability

Dimensions of Testing

There are many different dimensions to consider:

1.Layers of the application (database, APIs, UI)
2.Scale of testing (unit, module, integration, scenario)
3.Type of testing (functional, performance, security, etc.)
4.Methodology (exploratory, scripted manual, automated)

Stages of Testing

Unit Testing

During This first round of testing, the program is submitted to assessments that focus on specific units or components of the software to determine whether each one is fully functional. In this phase, a unit can refer to a function, individual program or even a procedure, and White box testing method is usually used to get the job done. One of the biggest benefits of this testing phase is that it can be run every time a piece of code is changed, allowing issues to be resolved as quickly as possible. It quite common for software developers to perform unit tests before delivering software to testers for formal testing.

Integration Testing

Integration testing allows individuals the opportunity to combine all of the units within a program and test them as a group. This testing level is designed to find interface defects between the modules/functions. This is particularly beneficial because it determines how efficiently the units are running together. Keep in mind that no matter how efficiently each unit is running, if they properly integrated, it will affect the functionality of the software program. In order to run these types of tests, individuals can make use of various testing methods, but the specific method that will be used to get the job done will depend greatly on the way in which the units are defined.

System Testing

System testing is the first level in which the complete application is tested as a whole. The goal at this level is to evaluate whether the system has complied with all of the outlined requirements and to see that it meets Quality Standards. System testing is undertaken by independent testers who haven’t played a role in developing the program. This testing is performed in an environment that closely mirrors production. System Testing is very important because it verifies that the application meets the technical, functional, and business requirements that were set by the customer.

Acceptance Testing

The final level, Acceptance testing (or User Acceptance Testing), is conducted to determine whether the system is ready for release. During the Software development life cycle, requirements changes can sometimes be misinterpreted in a fashion that does not meet the intended needs of the users. During this final phase, the user will test the system to find out whether the application meets their business needs. Once this process has been completed and the software has passed, the program will then be delivered to production. The extensiveness of these tests is just another reason why bringing software testers in early is important. When a program is more thoroughly tested, a greater number of bugs will be detected; this ultimately results in higher quality software.

Types of testing

Black box testing

It is also called as Behavioural/Specification-Based/Input-Output Testing. Black Box Testing is a software testing method in which testers evaluate the functionality of the software under test without looking at the internal code structure. This can be applied to every level of software testing such as Unit, Integration, System and Acceptance Testing.

White box testing

It is also called as Glass Box, Clear Box, Structural Testing. White Box Testing is based on applications internal code structure. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. This testing usually done at the unit level.

White Box Testing Techniques:

1.Statement Coverage

2.Branch Coverage

3.Path Coverage

Objectives of Testing

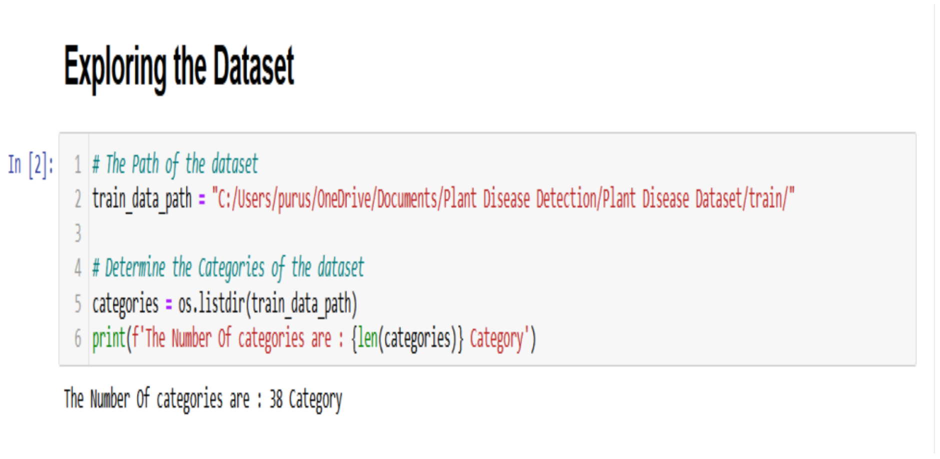
1.Validation of System Requirements: Ensure that the system meets all functional and non-functional requirements, such as prediction, accuracy, security, and usability.

2.Verification of Algorithm Performance: Test the implemented algorithms, including HDWOA, RFE, and the MLP Classifier, for feature selection and classification accuracy.

3.Identification of Bugs and Errors: Detect and resolve issues in the system, such as incorrect predictions, data handling errors, or UI responsiveness issues.

4.Performance Evaluation: Assess the system's ability to handle input data efficiently and produce results within acceptable time limits.

6. RESULTS

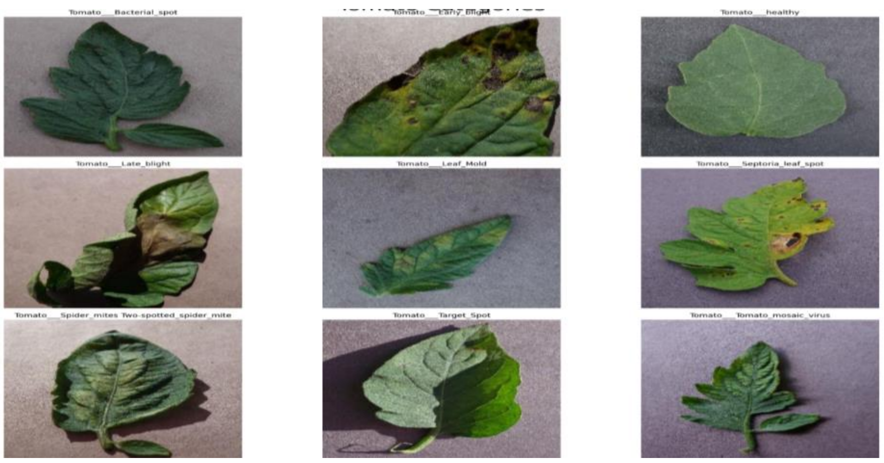


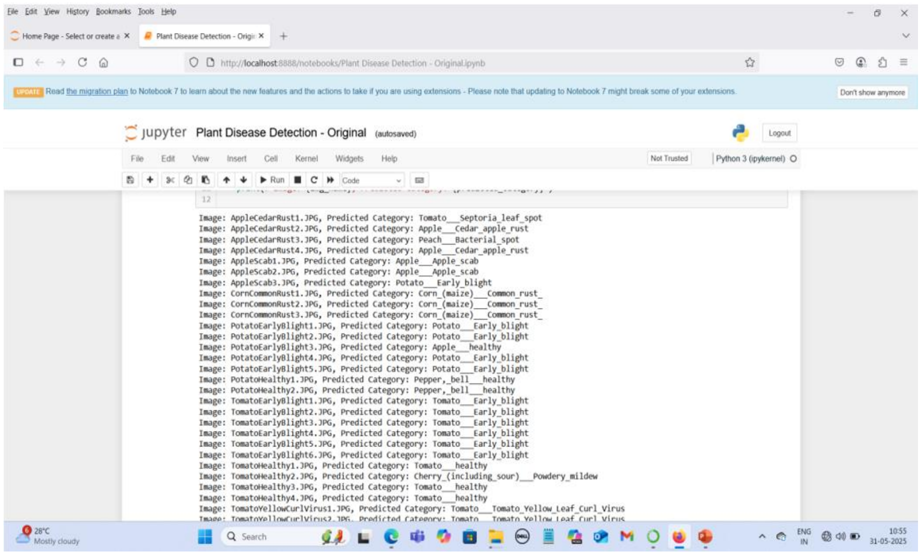
Exploring the Dataset

```
In [2]:
# The Path of the dataset
train_data_path = "C:/Users/purus/OneDrive/Documents/Plant Disease Detection/Plant Disease Dataset/train/"

# Determine the Categories of the dataset
categories = os.listdir(train_data_path)
print(f'The Number Of categories are : {len(categories)} Category')
```

```
The Number Of categories are : 38 Category
```




jupyter Plant Disease Detection - Original (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Logout

```
Image: AppleCedarRust1.JPG, Predicted Category: Tomato___Septoria_leaf_spot
Image: AppleCedarRust2.JPG, Predicted Category: Apple___Cedar_apple_rust
Image: AppleCedarRust3.JPG, Predicted Category: Peach___Bacterial_spot
Image: AppleCedarRust4.JPG, Predicted Category: Apple___Cedar_apple_rust
Image: AppleScab1.JPG, Predicted Category: Apple___Apple_scab
Image: AppleScab2.JPG, Predicted Category: Apple___Apple_scab
Image: AppleScab3.JPG, Predicted Category: Potato___Early_blight
Image: CornCommonRust1.JPG, Predicted Category: Corn_(maize)___Common_rust_
Image: CornCommonRust2.JPG, Predicted Category: Corn_(maize)___Common_rust_
Image: CornCommonRust3.JPG, Predicted Category: Corn_(maize)___Common_rust_
Image: PotatoEarlyBlight1.JPG, Predicted Category: Potato___Early_blight
Image: PotatoEarlyBlight2.JPG, Predicted Category: Potato___Early_blight
Image: PotatoEarlyBlight3.JPG, Predicted Category: Apple___healthy
Image: PotatoEarlyBlight4.JPG, Predicted Category: Potato___Early_blight
Image: PotatoEarlyBlight5.JPG, Predicted Category: Potato___Early_blight
Image: PotatoHealthy1.JPG, Predicted Category: Pepper,_bell___healthy
Image: PotatoHealthy2.JPG, Predicted Category: Pepper,_bell___healthy
Image: TomatoEarlyBlight1.JPG, Predicted Category: Tomato___Early_blight
Image: TomatoEarlyBlight2.JPG, Predicted Category: Tomato___Early_blight
Image: TomatoEarlyBlight3.JPG, Predicted Category: Tomato___Early_blight
Image: TomatoEarlyBlight4.JPG, Predicted Category: Tomato___Early_blight
Image: TomatoEarlyBlight5.JPG, Predicted Category: Tomato___Early_blight
Image: TomatoEarlyBlight6.JPG, Predicted Category: Tomato___Early_blight
Image: TomatoHealthy1.JPG, Predicted Category: Tomato___healthy
Image: TomatoHealthy2.JPG, Predicted Category: Cherry_(including_sour)___Powdery_mildew
Image: TomatoHealthy3.JPG, Predicted Category: Tomato___healthy
Image: TomatoHealthy4.JPG, Predicted Category: Tomato___healthy
Image: TomatoYellowCurlVirus1.JPG, Predicted Category: Tomato___Tomato_Yellow_Leaf_Curl_Virus
Image: TomatoYellowCurlVirus2.JPG, Predicted Category: Tomato___Tomato_Yellow_Leaf_Curl_Virus
```

7.CONCLUSION & FUTURE SCOPE

Conclusion

A robust crop disease detection system utilizing Convolutional Neural Networks (CNNs) offers a significant leap forward in agricultural practices. The project successfully demonstrates the feasibility and effectiveness of employing CNNs for accurate and early identification of various crop diseases.

This system has the potential to revolutionize how farmers manage their crops, moving from reactive treatments to proactive interventions. By providing timely and precise diagnoses, the system can help minimize crop losses, reduce the indiscriminate use of pesticides, and ultimately lead to healthier yields and more sustainable farming. The ability to identify diseases at early stages allows for targeted treatments, thereby optimizing resource allocation and reducing environmental impact.

Future Scope

This project can be expanded to support real-time disease detection through mobile apps, enabling instant diagnosis in the field. Integration with weather and soil data can further enhance prediction accuracy and aid in precision agriculture. The focus will be on enhancing accuracy and robustness through larger, more diverse datasets and advanced CNN architectures, including exploring few-shot and zero-shot learning for rapidly identifying new diseases. Crucially, future systems will prioritize integration with advanced technologies like the

Internet of Things (IoT) and drones for real-time, large-scale field monitoring, leveraging hyperspectral imaging for early disease detection, and integrating with agricultural robotics for automated, targeted interventions. This evolution will lead to sophisticated predictive analytics and decision support systems that can forecast disease outbreaks, optimize yields, and generate precise treatment maps, all within user-friendly platforms for farmers.

REFERENCES

- Mohanty, S. P., Hughes, D. P., & Salathé, M. (2023). Using deep learning for image-based plant disease detection. Frontiers in Plant Science, 7, 1419.

  DOI: 10.3389/fpls.2016.01419
- Zhang, C., & Lu, H. (2024). Attention mechanism in deep learning for plant disease detection: A survey. Computers and Electronics in Agriculture, 175, 105564.
  DOI: 10.1016/j.compag.2020.105564
- Zhang, Y., & Yang, Z. (2021). Multi-scale attention mechanism for plant disease recognition using deep learning. Biosystems Engineering, 198, 39-49.

  DOI: 10.1016/j.biosystemseng.2020.11.013