International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 9, Issue 4, pp 119-126, April-2025
https://doi.org/10.47001/IRJIET/2025.904018

Signify: An ML Based Plant Disease Detection System

[1]Neha Bagul, [2]Sayali Chorghe, [3]Riya Maji, [4]Diptimai Sahoo, [5]Manisha Hatkar

[1,2,3,4]Student, Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India
[5]Professor, Dept. of AI&ML, Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India

***Abstract* - Agriculture remains the backbone of many economies, and plant health is essential for food security and high yields. Traditional methods of plant disease identification are slow, inconsistent, and inaccessible for many farmers. To address these challenges, we propose a deep learning-based Plant Disease Detection System that identifies plant diseases through image recognition. Users can upload images of diseased leaves to receive fast, accurate diagnoses and tailored treatments. Utilizing transfer learning, our system fine-tunes the VGG-16 Convolutional Neural Network (CNN) on the Plant Village dataset. The web-based interface, built using Flask, enables easy interaction and disease management. This paper discusses the development and implementation of the system, highlighting its potential to revolutionize plant disease management and support sustainable agriculture. The approach is validated through rigorous performance metrics, and future enhancements are explored.**

Keywords: Leaf Disease Detection, Machine Learning, VGG-16, Transfer Learning, Image Classification, Deep Learning, Smart Agriculture, Plant Village Dataset, Precision Farming, Convolutional Neural Networks.

I. INTRODUCTION

The growing global population and changing food consumption patterns have significantly increased the demand for agricultural productivity. Plant diseases are a major threat to this goal, causing up to 40% leaf losses annually, as reported by the FAO. Early detection and timely treatment can mitigate these losses, making disease diagnosis a vital element of sustainable agriculture.

Traditional diagnosis relies heavily on expert intervention, which is often unavailable in rural settings. These methods are time-consuming, inconsistent, and error-prone. In contrast, the application of artificial intelligence (AI) and computer vision in agriculture provides promising alternatives. Deep learning, particularly convolutional neural networks (CNNs), enables automated image-based disease detection with high accuracy. This study presents a deep learning-based system leveraging the VGG-16 model to identify leaf diseases effectively. The platform offers a user-friendly web interface and supports real-time disease identification and remedy suggestions.

The agriculture sector is witnessing a digital transformation, with technologies such as the Internet of Things (IoT), drones, and AI enabling smarter farming solutions. Automated disease detection fits well within this ecosystem, offering real-time insights and reducing dependency on manual field surveys. By enabling early intervention, such systems contribute to reduced pesticide use, better resource allocation, and improved yield.

In developing countries, where access to expert agronomists is limited, AI-based tools can play a crucial role. Empowering farmers with simple and intuitive disease detection tools will enable them to take proactive steps in plant protection, thereby enhancing leaf health and productivity.

II. RELATED WORK / LITERATURE SURVEY

Early research on plant disease detection relied on basic image processing techniques like edge detection and histogram analysis. Kulkarni et al. [1] used these methods, but their approach lacked adaptability under varying environmental conditions. Jaware et al. [2] employed segmentation techniques to isolate affected regions but lacked real-time processing.

The emergence of CNNs marked a turning point in plant disease classification. CNNs can learn hierarchical features, making them ideal for visual recognition tasks. Noteworthy commercial efforts include the Plantix app, which provides disease diagnosis via smartphones but is proprietary and limited in scope.

Hughes and Salathe [5] introduced an open-source repository to facilitate AI-driven plant disease detection using deep learning. Their work enabled the training of various CNN models, providing the foundation for further research. Recent studies using models such as ResNet, DenseNet, and InceptionV3 have demonstrated improved accuracy and robustness in disease classification tasks.

Transfer learning using pre-trained models such as VGG-16 and ResNet50 has proven to enhance performance even with limited domain-specific data. These models, trained on ImageNet, can be fine-tuned for agricultural datasets, reducing training time while improving accuracy.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 9, Issue 4, pp 119-126, April-2025
https://doi.org/10.47001/IRJIET/2025.904018

A comparative study of different architectures revealed that although newer models like Efficient Net may achieve higher accuracy, VGG-16 remains a strong baseline due to its simplicity, reduced computational overhead, and consistent performance in resource-constrained environments.

The limitations of these approaches highlight the need for a robust, accurate, and scalable solution. Transfer learning, especially with models like VGG-16 and ResNet50, has emerged as a promising strategy. These models, pre-trained on massive datasets like ImageNet, can be fine-tuned for specific tasks with relatively small domain-specific datasets, thus reducing training time and improving performance.

III. METHODOLOGY

3.1 Dataset and Model Development

3.1.1 Dataset and Preprocessing

The Plant Village dataset contains over 61,000 images across 39 categories of healthy and diseased plant leaves. These categories include leaves like tomato, potato, apple, corn, grape, and others with both healthy and multiple diseased states. Images were resized to 224x224 pixels to match the input requirements of VGG-16.

Fig 3.1.1 Sample Distribution of Dataset

Data augmentation techniques such as rotation, flipping, zooming, shearing, and brightness adjustment were used to increase dataset diversity and model generalization. Normalization of pixel values (scaling between 0 and 1) ensured that the input data was consistent with VGG-16's expected input range. The dataset was divided using an 80/20 split into training and testing sets, and stratified sampling was used to ensure balanced representation of each class.

Image quality enhancement filters like Gaussian blur, sharpening, and contrast normalization were applied to reduce noise and make the model robust to field images captured under varying lighting and focus conditions.

3.1.2 Transfer Learning Using VGG-16

VGG-16 is a 16-layer deep CNN architecture originally designed for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). For our use case, the final fully connected layer was replaced to support classification across 39 disease categories. The rest of the architecture was retained for feature extraction.

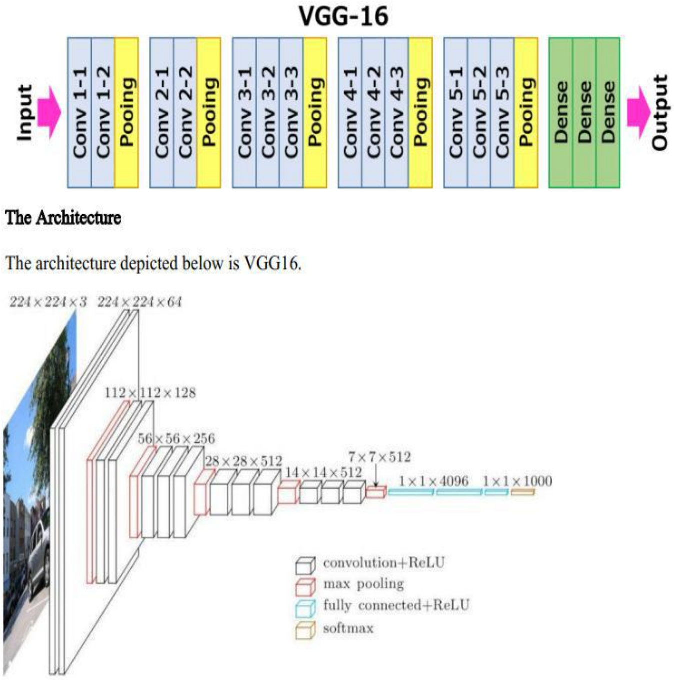

Fig 3.1.2 CNN (VGG16) Model Architecture

Transfer learning allows the model to retain generic features such as edges and textures from the original dataset while learning disease-specific features through fine-tuning. The model was trained using the PyTorch framework with the Adam optimizer, an initial learning rate of 0.0001, and Cross Entropy loss. Training was carried out on an NVIDIA GPU-based system.

To prevent overfitting, we implemented dropout layers and used early stopping based on validation loss. Batch normalization further stabilized and accelerated training. K-fold cross-validation was employed to assess model robustness and avoid data bias.

3.1.3 Web Interface Integration

A key component of our system is its accessibility via a web interface built using Flask. The interface enables users to:

- Upload leaf images directly from a mobile or desktop device
- Get instant disease detection results
- View recommended treatments or fertilizers based on the identified disease

The system also logs user data anonymously for future model improvement through feedback loops. A built-in database

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 4, pp 119-126, April-2025

https://doi.org/10.47001/IRJIET/2025.904018

stores disease-specific remedies, curated from agricultural experts and verified government sources.

The user interface supports regional languages and includes text-to-speech features for illiterate users. The backend integrates a lightweight REST API for scalability and future mobile app support.

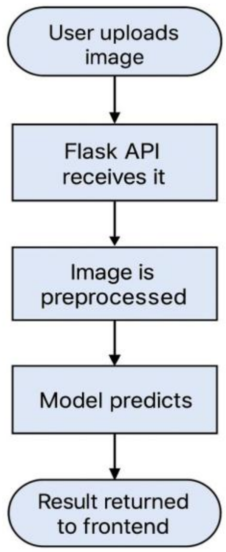

Fig 3.1.3 Process Flow Diagram

IV. IMPLEMENTATION

4.1 Development and Tools

Machine Learning & Model Development

The core of the system is built using PyTorch, with a VGG16 architecture applied via transfer learning. The model is trained on a labeled dataset of plant leaf images, and the final model is saved as a .pt file.

To assist in image preprocessing and model training, the project uses:

- Torchvision: for image transformation and dataset loading.
- NumPy: for numerical operations.
- PIL(Pillow) : for handling and converting images.

The model is trained separately (outside the Flask app) and loaded during runtime for inference.

Backend Development

The backend is powered by Flask, a lightweight Python web framework. Flask manages the routing, file uploads, and interaction with the model. Key backend elements include:

- app.py: The main Flask application that defines routes for uploading images and displaying results.
- utils.py: A helper script used for image processing before making predictions.
- plant_disease_model_1.pt: The pre-trained PyTorch model used for inference.
- Flask-WTF: Used for handling forms and enabling CSRF protection.
- Werkzeug: A dependency under Flask used for WSGI utilities.

When a user uploads an image, the backend preprocesses it, feeds it to the model, and returns the predicted class.

Frontend Development

The frontend is composed of basic HTML and CSS to provide a clean interface for users. The HTML templates (inside the templates/ folder) are rendered using Flask's Jinja2 engine. Static assets like stylesheets and images are stored in the static/ directory.

Users can upload plant leaf images using a simple web form, and the prediction result is displayed on the same page.

Testing & Execution

The application is tested and run locally using Flask's development server. Jupyter Notebook was used during the model development phase.

The project is modular and easy to test:

- Run the Flask app locally after installing dependencies.
- Upload an image via the web interface and view the result.

4.2 System Architecture

The system is designed using a modular architecture that clearly separates the concerns of frontend, backend, and model training. The overall flow is as follows:

1. Frontend Module

The frontend is responsible for user interaction and visual display, and it is built using HTML, CSS, JavaScript, and Flask for deployment. It includes several pages, starting with a Home Page that provides an introductory interface with a description of the project and its purpose, a Plant Leaf Detection Page that allows users to upload images of plant leaves for analysis, and a Contact Us Page offering team details or a support form. Users can upload images through a form interface, which are then sent to the backend for processing via an API.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 4, pp 119-126, April-2025

https://doi.org/10.47001/IRJIET/2025.904018

2. Backend Logic

The backend serves as the intermediary between the interface is user-friendly and connects seamlessly with the machine learning model, which has been carefully developed for accurate disease detection using the Flask framework in Python. Flask acts as a lightweight server that handles HTTP requests and manages routing between the frontend and the model logic. Its core functionality includes receiving uploaded images from the frontend, preprocessing these images through resizing, normalization, and other techniques, passing the processed the image is sent to the machine learning model for prediction, and the results are then returned and displayed on the frontend.

3. Machine Learning Model Module

This module handles the training and evaluation of the plant leaf disease classification model and is separated from the deployment, we designed the backend to follow a clean and modular architecture. The model training process starts with preparing the dataset preparation, which includes organizing the image data into distinct classes such as tomato, potato, strawberry, and grape. Approximately 40 images per class are used for training, with a similar distribution maintained for the testing dataset to ensure balanced evaluation.

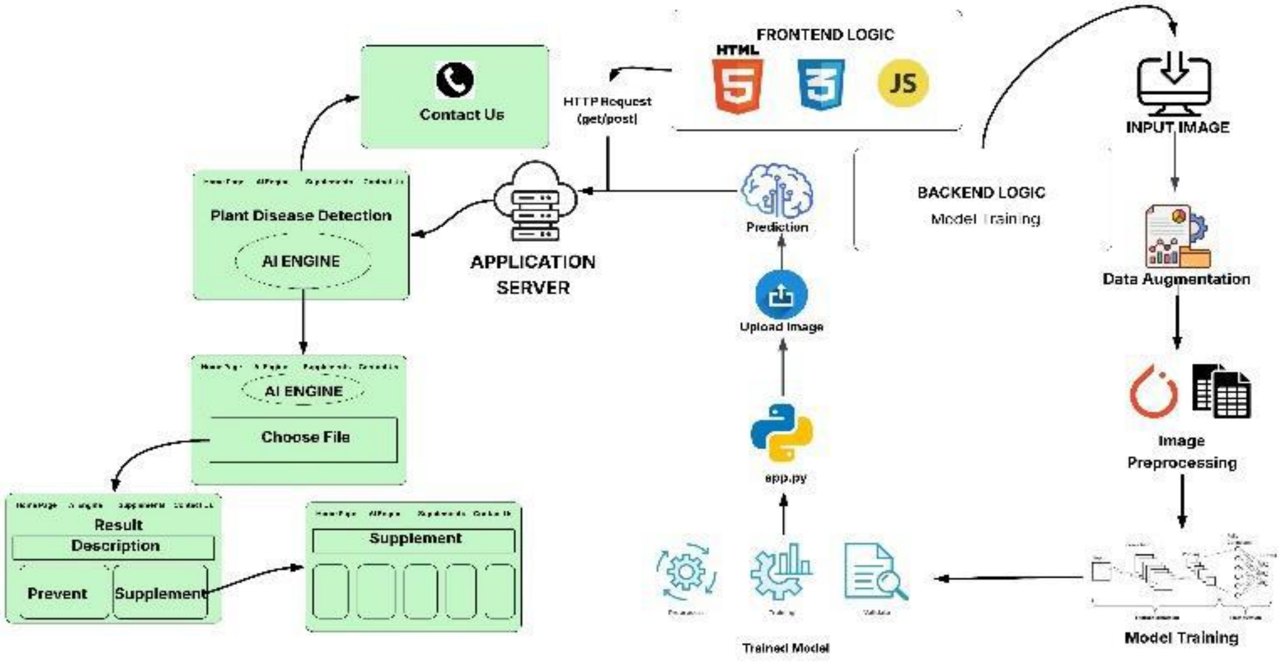

Fig 4.2.1 System Architecture Diagram

4. Data Augmentation and Model Architecture

The system incorporates data augmentation techniques such as rotation, flipping, zooming, and brightness adjustment to make the system more reliable and better at handling different kinds of data model. The model architecture is based on transfer learning using pre-trained convolutional neural networks like VGG16 and MobileNetV2, with custom classification layers added on top to adapt to the specific plant disease classification task. The training process was done using Jupyter Notebook, which made it easier to test and twist the model step by step utilizing GPU acceleration for faster computation. Upon completion, the trained model was exported as a .pt file to facilitate smooth integration into the Flask server.

5. Application Server

The application server, built using Flask, acts as a middleware between the frontend and the trained model, exposing API endpoints such as /predict to accept plant leaf images via POST requests and return the predicted disease class, and an optional /status endpoint for server health checks or debugging purposes.

6. Deployment

Finally, the deployment layer involves integrating both the frontend and backend along with a unified Flask dashboard, allowing users to interact with the model in real-time for live disease predictions.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 4, pp 119-126, April-2025

https://doi.org/10.47001/IRJIET/2025.904018

4.3 Key Mathematical Formulae

Evaluation metrics

Accuracy, Precision, Recall, and F1 Score , play a crucial role in assessing the performance of classification models, particularly in real-world scenarios where data imbalance is common.

Accuracy - measures the overall correctness of the model but can be misleading when one class dominates the dataset. In such cases, a high accuracy might simply reflect the model's bias toward the majority class.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

Precision - indicates how many of the predicted positive cases were actually correct, which is essential when false positives have significant consequences, such as misdiagnosing a healthy leaf as diseased.

$$Precision = \frac{TP}{TP + FP}$$

Recall / sensitivity - measures the model's ability to correctly identify all actual positive cases, making it critical in minimizing false negatives—important when undetected diseases could spread further.

$$Recall = \frac{TP}{TP + FN}$$

F1 Score - provides a balanced measure by combining precision and recall, offering a more reliable indicator of performance when classes are imbalanced. Together, these metrics help ensure the model is not only accurate but also dependable across varying conditions and class distributions.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Convolutional Neural Network

A Convolutional Neural Network (CNN) based on the pre-trained VGG16 architecture is used to classify plant leaf images into disease categories. CNNs are highly effective for image processing tasks as they automatically extract key features like color, shape, and texture through layers of convolution, activation (ReLU), and pooling. By leveraging transfer learning, the model uses VGG16's deep feature extraction capabilities and fine-tunes the final classification layer to detect specific plant diseases with high accuracy, making it well-suited for visual pattern recognition in agricultural diagnostics.

CNN Layer formula for basic computation in a conv layer:

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

Where:

O: output dimension
W: input width/height
F: filter/kernel size
P: padding
S: stride

V. FUTURE SCOPE / LIMITATIONS

5.1 Future Work

The Plant Disease Detection System, while effective in its current form, has vast potential for further development and deployment at a larger scale. Some of the major areas for future improvement and innovation include:

- **Dataset Expansion:** Incorporate more plant species, regional varieties, and rare diseases. Curating localized datasets will improve the system's adaptability across different agro-climatic zones.
- **Integration with Drones and IoT Devices:** Enable large-scale and real-time monitoring of leaves through drone imagery and IoT sensors. This approach makes it easier to detect diseases at an early stage disease outbreaks over wide agricultural lands.
- **Offline Functionality:** Develop a native mobile application with offline capabilities. This would help farmers in regions where limited internet access to utilize the system seamlessly.
- **Multilingual Support and Accessibility:** Include support for multiple regional languages, text-to-speech features, and voice-based inputs to cater to a broader demographic of farmers, including those who are illiterate.
- **Advanced Deep Learning Models:** Experiment with ensemble learning, Vision Transformers, and newer CNN architectures to improve classification accuracy and model interpretability.

5.2 Model Limitations

While the plant disease detection system offers significant potential for agricultural diagnostics, several limitations need to be considered for its effective deployment and usage in real-world environments. These constraints

ISSN (online): 2581-3048
Volume 9, Issue 4, pp 119-126, April-2025
https://doi.org/10.47001/IRJIET/2025.904018

impact the system's overall accuracy, scalability, and adaptability. Understanding these limitations is crucial to refining the technology and addressing challenges in diverse agricultural settings. Below are the key limitations of the current system:

Limited Disease Coverage: The model is trained only on the diseases available in the Plant Village dataset. It cannot recognize new, rare, or unlisted diseases, which limits its diagnostic capability in diverse agricultural contexts.

Dependence on Image Quality: The accuracy of disease The accuracy of detection largely depends on how clear and well-lit the uploaded image is images. Blurry, poorly lit, or background-cluttered images can lead to incorrect or failed predictions.

Internet Dependency: The current system requires internet connectivity to access the web-based application. This poses a challenge for farmers in rural or remote areas with limited or no internet access.

Lack of Real-time Field Integration: The system does not yet connect with real-time farming data from drones, smart sensors, and other IoT devices, or satellite feeds, limiting its ability to monitor large-scale farms or detect early-stage disease symptoms dynamically.

No Self-Learning Mechanism: The system currently lacks an automated feedback or learning loop. It does not improve itself over time based on user corrections or field results, which can hinder long-term adaptability and intelligence.

VI. RESULTS AND DISCUSSIONS

The trained VGG-16 model achieved an overall classification accuracy of 95.2% on the test dataset. The precision was calculated at 94.5%, recall at 95.0%, and F1-score at 94.7%. These metrics indicate that the system is highly effective in distinguishing between various disease types.

We compared the model's performance with other architectures like ResNet-50 (accuracy 96.1%), InceptionV3 (95.4%), and MobileNetV2 (93.7%). While ResNet-50 showed marginally better performance, VGG-16 was selected for its optimal balance between accuracy and computational requirements, especially for deployment in low-resource settings.To interpret model predictions, we used Gradient-weighted Grad-CAM (Class Activation Mapping) is used to highlight the parts of the image that had the biggest impact on the classification decision, helping us visualize which areas of the image were most important for the model's prediction. This enhanced transparency and allowed us to validate the relevance of predictions.

Before diving into the main phase of the plant disease detection project, we took the time to carefully evaluate several different approaches. This allowed us to choose the most effective model for our needs project, we conducted a comprehensive evaluation of multiple Convolutional Neural Network (CNN) architectures to identify the most suitable model in terms of performance, efficiency, and practicality. Specifically, we compared a lightweight Custom CNN, the widely adopted VGG16, and the deeper, more complex ResNet50, focusing on key metrics such as accuracy, precision, recall, training time, and parameter count. This comparison aimed to balance the trade-off between computational cost and classification performance. The results demonstrated that while ResNet50 achieved the highest accuracy and precision due to its deeper residual connections and advanced feature extraction, it also required significantly more computational resources. VGG16 offered strong performance with moderate complexity, whereas the Custom CNN proved advantageous for quick training and deployment in resource-constrained environments. These insights informed our approach to selecting the right model highlighted how crucial it is to balance performance with practical considerations, especially when working with limited training data transfer learning in achieving high accuracy even with limited training data.

Table 1: Analysis of various model

	Custom CNN	VGG16	ResNet50
Accuracy	92.5%	94.8%	96.2%
Precision	91.2%	94.5%	96.1%
Recall	92.0%	94.2%	95.9%
Training Time	Fast (~4 mins)	Moderate	Slowest
Parameters	~0.5M	~14.7M	~23.5M
Comments	Lightweight, easy to train	Strong performance, low overfitting	Highest accuracy, but heavy

This work shows that even with limited training data, deep learning models, especially those using transfer learning, can classify agricultural diseases effectively. The use of pretrained models like ResNet and VGG significantly improved performance due to their generalized feature extraction capabilities.

Overfitting Behavior: Custom CNN showed mild overfitting also VGG16 and ResNet50 generalized better due to transfer learning.

ISSN (online): 2581-3048
Volume 9, Issue 4, pp 119-126, April-2025
https://doi.org/10.47001/IRJIET/2025.904018

Data Augmentation Effects: Rotation and flipping improved generalization. Moreover, overuse led to noise and slower convergence.

Robustness Test: Slightly blurred images were more accurately handled by ResNet50 and VGG16.

VII. CONCLUSION

This paper presents a robust, scalable, and user-friendly deep learning-based Leaf Disease Detection System, designed to aid farmers in early and accurate disease identification. Leveraging the power of transfer learning and the VGG-16 architecture, the system delivers high performance with minimal training data and computational requirements.

By combining a powerful backend with an intuitive frontend, the solution bridges the gap between cutting-edge AI and real-world usability. It offers significant benefits in reducing leaf loss, minimizing dependency on manual inspections, and promoting smart agricultural practices.

Despite promising results, limitations remain. The model is restricted to the disease classes present in the training dataset. Also, variations in image capture conditions (e.g., background clutter, multiple leaves) may affect performance. Future work will focus on:

- Expanding the dataset to include more leaves and localized disease variants.
- Integrating drone and IoT-based image feeds for real-time monitoring.
- Building an Android-based offline version for remote areas.
- Using ensemble models and meta-learning for improved generalization.
- Developing a collaborative platform for farmers to share observations and feedback.

This system marks a significant step forward in the fusion of AI and agriculture, and with continued research and community engagement, it holds great promise for enhancing food security and sustainable farming.

ACKNOWLEDGEMENT

The successful completion of any task holds little meaning if we fail to recognize the individuals whose efforts and support made it possible. Without their valuable contributions, this project would have remained just an idea.

First and foremost, we express our sincere gratitude to Prof. Sonali Deshpande, Head of the Computer Science & Engineering (AI & ML) Department, along with all the esteemed faculty members of our department. Their consistent guidance, motivation, and support were instrumental to the success of the project. We are especially thankful to our internal guide, Prof. Manisha Hatkar, whose dedication, insight, and encouragement greatly inspired us. Her expert advice and availability whenever we needed help were invaluable to our progress.

We would also like to acknowledge and appreciate all those who, directly or indirectly, contributed to the successful execution of this major project. Their assistance and guidance played a crucial role in helping us bring this endeavour to life.

Lastly, and most importantly, we extend our heartfelt thanks to our parents and well-wishers. Their unwavering support and encouragement have been a constant source of strength throughout this journey.

REFERENCES

[1] A.H. Kulkarni, A.R.K. Patil, "Applying image processing technique to detect plant diseases," International Journal of Modern Engineering Research, vol. 2, no. 5, pp. 3661–3664, 2012.
[2] T.H. Jaware, R.D. Badgujar, P.G. Patil, "Leaf disease detection using image segmentation," National Conference on Advances in Communication and Computing, pp. 190–194, 2012.
[3] S.B. Dhaygude, N.P. Kumbhar, "Agricultural Plant Leaf Disease Detection Using Image Processing," IJAREEIE, vol. 2, no. 1, pp. 599–602, 2013.
[4] K. Simonyan, A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv preprint arXiv:1409.1556, 2015.
[5] D.P. Hughes, M. Salathe, "An open access repository of images on plant health to enable the development of mobile disease diagnostics," arXiv preprint arXiv:1511.08060, 2015.
[6] Mohanty, S.P., Hughes, D.P., & Salathé, M. (2016).Using deep learning for image-based plant disease detection. Frontiers in Plant Science, 7, 1419.
[7] Ferentinos, K.P. (2018). Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture, 145, 311–318.
[8] Barbedo, J.G.A. (2013). Digital image processing techniques for detecting, quantifying and classifying plant diseases. SpringerPlus, 2(1), 660.
[9] Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D.(2016). Deep neural networks based recognition of plant diseases by leaf image classification. Computational Intelligence and Neuroscience, 2016.
[10] Picon, A., Alvarez-Gila, A., Seitz, M., Ortiz-Barredo, A., Echazarra, J., & Johannes, A. (2019). Deep

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 9, Issue 4, pp 119-126, April-2025

https://doi.org/10.47001/IRJIET/2025.904018

convolutional neural networks for mobile capture device-based leaf disease classification in the wild. Computers and Electronics in Agriculture, 161, 280–290.

[11] Zhang, S., Wu, X., & You, Z. (2017).Leaf image based cucumber disease recognition using sparse representation classification. Computers and Electronics in Agriculture, 134, 135–141.

[12] Amara, J., Bouaziz, B., & Algergawy, A.(2017). A deep learning-based approach for banana leaf diseases classification. In Datenbanksysteme für Business, Technologie und Web (BTW 2017) – Workshopband (pp. 79–88).

[13] Too, E.C., Yujian, L., Njuki, S., & Yingchun, L. (2019).A comparative study of fine-tuning deep learning models for plant disease identification. Computers and Electronics in Agriculture, 161, 272–279.

AUTHORS BIOGRAPHY

Neha Bagul,
Pursuing B.E.in CSE - AIML (Final year) at Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India.

Sayali Chorghe,
Pursuing B.E.in CSE - AIML (Final year) at Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India.

Riya Maji,
Pursuing B.E.in CSE - AIML (Final year) at Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India.

Diptimai Sahoo,
Pursuing B.E.in CSE - AIML (Final year) at Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India.

Manisha Hatkar,
Professor, Department CSE (AIML) at Smt. Indira Gandhi College of Engineering, Ghansoli, Navi Mumbai, Maharashtra, India.

Citation of this Article:

Neha Bagul, Sayali Chorghe, Riya Maji, Diptimai Sahoo, & Manisha Hatkar. (2025). Signify: An ML Based Plant Disease Detection System. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 9(4), 119-126. Article DOI https://doi.org/10.47001/IRJIET/2025.904018

****