Detection of Leaf Disease in Plantation Process for Fruits, Vegetables, Grains and Cereals using Application

Madhuri Kanojiya[1], Lokesh Chouhan[2], Vipin Tiwari[3,*], Dheresh Soni[4], Devika A. Verma[5], Yashwant Dongre[5]

[1]National Institute of Food Technology Entrepreneurship and Management, Delhi, India

[2]National Forensic Sciences University, Goa, India

[3]Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, India

[4]School of Computing Science and Engineering, VIT Bhopal University, Bhopal, India

[5]Vishwakarma Institute of Information Technology, Pune, India

Emails: dr.madhurikanojiya@gmail.com; lokesh.chouhan@nfsu.ac.in; vipintiwari1@gmail.com; research.dheresh@gmail.com; devika.verma@viit.ac.in; yashwant.dongre@gmail.com

Abstract

One of the most important sectors for providing for daily human requirements is agriculture. At the same time, digitization has a big impact on a number of businesses, making it simpler to carry out a number of challenging tasks. In order to help the farmer and the consumer, technology and digitization must be adopted. Utilizing technology and routine monitoring, diseases can be identified and eliminated, increasing agricultural output. This paper suggests a system for recognizing and categorizing plant illnesses, initially focused on five separate classes: two fruit classes, one vegetable class, one edible pulse class, and one-grain class. The Plant Village and UCI ML Repository Dataset, which is well known as a freely accessible, accepted standard, and reliable data source, was used for this purpose. Based on them, a CNN model is prepared for analyzing them with an accuracy upto 95.42%. Image segmentation will also play a role in calculating precise amount of infection followingly, a good interface is must to utilize it in a proper way for a user which can be provided in the form of app, a feature that every user requires on daily basis.

Keywords: Agriculture; Digitization; Plant Village; CNN model; Image Segmentation

1. Introduction

Early plant disease detection is now a top priority in the sector of agriculture in order to reduce loss and boost healthy yield. Similar to animal diseases, plant diseases can have either an infectious or non-infectious. Depending on the cause, location, and type of the impact site, its symptoms [1] may vary. Abiotic and biotic factors both exist. Plant diseases manifest as wilting, spotting (necrosis), mould, pustules, rot, hypertrophy and hyperplasia (over-growth), deformation, discoloration, and tissue dis-integration. We need to be aware of the particular pesticide and the recommended dosage based on these numerous variances. Improper issue detection causes significant harm to farmers as well as health problems for consumers. This article aims at covering such aspect with the modern-day solution, i.e., of image processing and computer vision along with deep learning to get the precise output. A good interface with it makes the project a user-friendly and beneficial to use as wide usage of app is coming into picture and would be best if assistance can be provided in this form. Application will be capable of detecting five different classes, i.e., Cherry (Healthy & Powdery Mildew), Tomato (Healthy, Mosaic virus, Early/Late Blight, Septoria leaf spot, Yellow leaf curl virus, Leaf mould, Spotted spider termites, Bacterial spot), Potato (Healthy, Early/Late Blight), Corn (Healthy, Commun Rust, Cercospora Grey spot, Northern leaf Blight) & Rice (Smut, Brown spot, Bacterial leaf Blight).

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

Rest of the article is structured as follows. The section 2 represents the related work in the field. Section 3 explains detailed design and implementation of proposed application. Section 4 discusses the result and outcome of article and section 5 finally conclude the article and displays the possible directions for future work.

2. Related Work

This section plays a major role in this work since all the resources required for the work are well analyzed and subjected to implementation. The paper [2] reviews the use of imaging techniques and computer vision for plant disease identification, focusing on image acquisition, preprocessing, segmentation, feature extraction and classification. It highlights the current trends and challenges in effectively detecting and managing plant diseases. The work presented by Noon et. al. [3] displayed an improved method for detecting crop and fruit leaf diseases under real-field conditions. The study [4] reviews 1349 papers from major databases, narrowing down to 176 relevant studies. The research highlights the use of hyperspectral imagery and vision-centered approaches for crops like grapes, rice and tomatoes. While SVMs and logistic regression showed high accuracy, challenges such as disease localization and model standardization remain. There is a need for more efficient, scalable models.

A. Choosing the CNN:

Starting with building the model, several traditional machine learning methods were analyzed, followed by CNN (Deep Learning method) which proves that manual feature extraction & classification [5][6] is not required as it can learn automatically on training & produce the accurate output. Keras (Tensorflow) module of python has such method. Model built on this forms the 80% of the backend for the project. CNN Architecture consists of seven layers [7], which are Input layer, Convolutional Layer, Pooling Layer, Fully Connected Layer, Softmax Layer, Activation Function & Output layer as shown in Fig. 1. For training the model efficiently to obtain a highly accurate output based, we need to choose the parameters wisely in these layers like learning rate, optimizers etc. Survey suggests that Adam optimizer [8] proves to be best among other optimizers [9] because it is faster and has minimum convergence compared to other optimizing techniques thus reducing training cost and prediction error much lessor with respect to number of iterations/epochs.

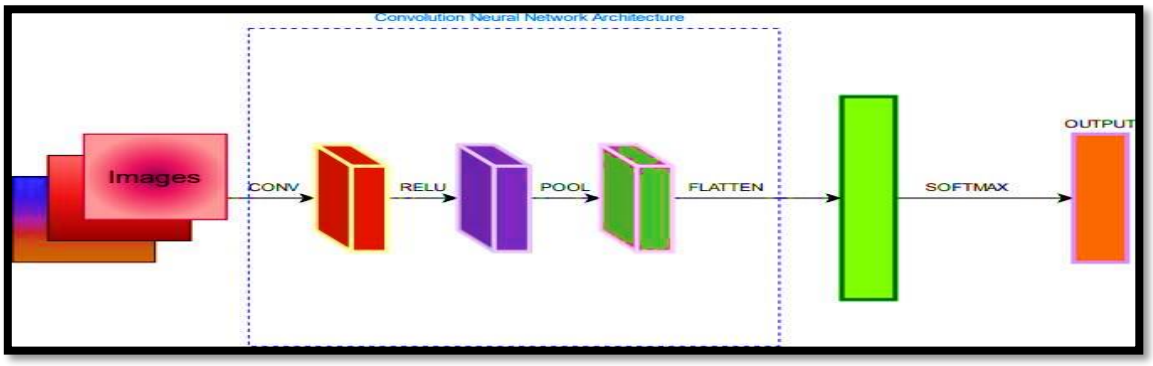


Figure 1. Layers of CNN Architecture

There are some pre-trained CNN Architectures such as VGG16, VGG19, ResNet50, Inception V3 etc which can be used for the same. Table 1 shows some of the related works on building model based on plant disease detection.

Table 1: Some related works in same domain

Work	Method	Average Accuracy
Mohit et. al. [10] Tomato Leaves Disease Detection (2019)	VGG16, InceptionV3 (GoogleNet) and Mo- bileNet.	91.2%
Liu et. al. [11] Grape Leaves Dis- ease Detection (2019)	SVM & RF	95.83%
Ramcharan et. al. [12] Cassava Dis- ease detection	Inception V3	93%
Kumari et. al.[13] Leaf Disease Iden- tification (2019)	K-Means & ANN	92.5%

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

Chen et. al. [14] Plant Disease Identification	INC VGGN	92%
Mercelin et. al. [15] Plant Disease Detection & Clas- sification	Pre-trained CNN	87%
Jiayue et. al. [16] Tomato fruits dis- eases detection	YOLOv2 CNN	97%
Shrivastava et. al. [17] Rice plant disease detection	Pre-trained CNN	91.37%
Priya et. al.[18] Nine types of dis- ease identification	VGG16, ResNet50, In- ceptionV3, Inception- ResNetV2 & Xception	87%
Manmohan et. al. [19] leaf and fruit disease detection	SVM classifier	91.73%
Ahmed et. al. [20] Tomato leaf diseases	CNN	84.49%
Abisha et. al. [21] Brinjal leaf diseases	RBFNN	87%
Liu et. al. [22] Apple leaf diseases	YOLOV5	92.7%

B. Data Augmentation technique:

It is nothing but increasing the dataset by producing imagination [23] alternatives to one image to gain a good understanding on each. In simple words, imagining the same in different scenarios. It plays a role a training the model to reduce the underfitting problem (which leads to misunderstanding of data) and increase the model's accuracy. It involves few important Geometric transformation processes such as Flipping, Cropping, Color space, Rotations & Translations.

C. Edge Detection:

It is widely utilized for various processes like features extraction, segmentation, background removal, image sharpening etc. There is a requirement for this image processing technique [24] to calculate the amount of infection in our project. In order to do so, firstly image must be segregated from the background, which can be done by detecting the edges (depending on continuity) and obtaining the leaf. Examples of Edge Detection algorithms are Canny, Prewitt, and Roberts & Sobel.

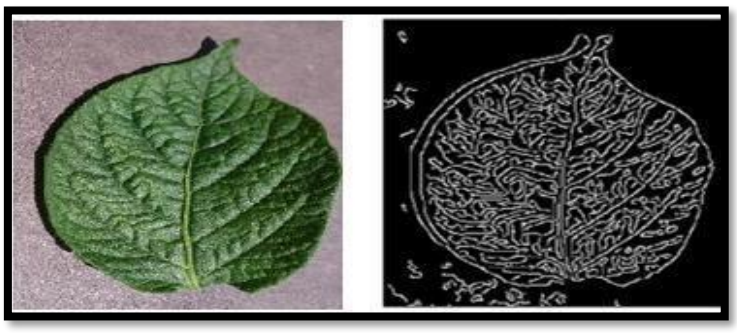


Figure 2. Edge Detection of a Leaf

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

D. Thresholding techniques:

Followed by Edge Detection, the Thresholding technique, which is a part of image segmentation [25], is necessary to implement to calculate infection percentage. It even plays a role in detecting the invalid image. Image will be binarized (converting into black and white image) depending on pixel intensities and set threshold value.

Otsu's algorithm here determines the threshold value automatically. It is obtained by processing the input image and then obtaining its histogram containing distribution of pixels because of which value is set for binarizing the image. Iterative threshold selection is method in which Otsu's algorithm can be initialized to threshold image, where each pixel is segmented into a class depending on a posteriori probability where the conditioned implicates that it belongs to class [26] where it is maximum.

E. Cloud Computing:

It is widely used services, which delivered through internet that includes storage, software, application, servers or database, which in turn eliminates the much of harder use. Advantages can be the Flexibility, Scalability, Cost & Ease of Access through remote any time when needed. There are three types of service models in cloud computing, i.e., Infrastructure as a Service (SaaS) for client's storage, database, servers etc., Platform as a Service (PaaS) [27] for building applications & Software as a Service (SaaS) for software to users without high hardware requirements.

Our applications demands the usage of PaaS model in order to run the backend interface to it's android frontend. For this, a webapp can be designed using a Flask – Python's Framework, to integrate with trained model backend and act as application's backend in cloud.

3. System Design and Implementation

A good literature survey brought a lot of clarity on user's mentality towards the application and requirements of the project, which will productively produce accurate outputs. A system is designed in such a way that user does not need much research & expertise to identify the problem. Starting with the Application Design Model (as shown in Fig. 3), which we can also consider as overall app flowchart, is the illustration of how app will work.

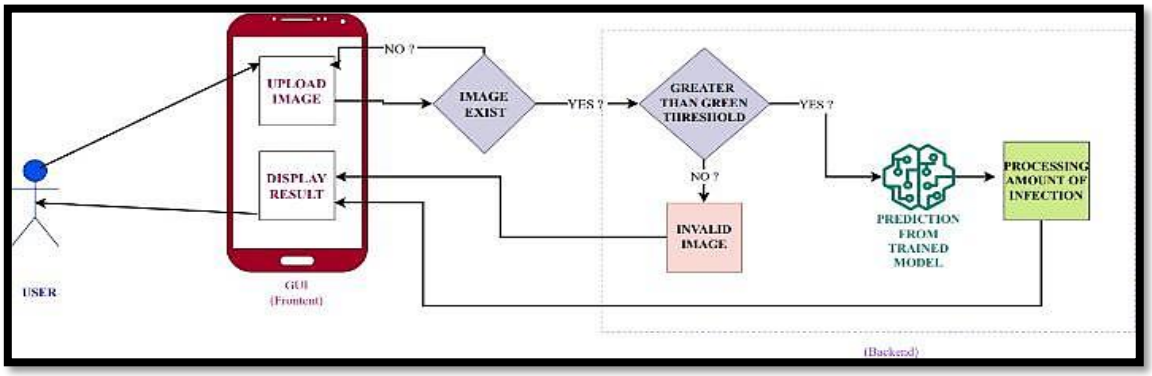


Figure 3. Application Design Mode

Application's backend consists of trained model, web-app running in cloud. Fig. 4 demonstrates the flow of backend in app. Internet connectivity is must to run the app.

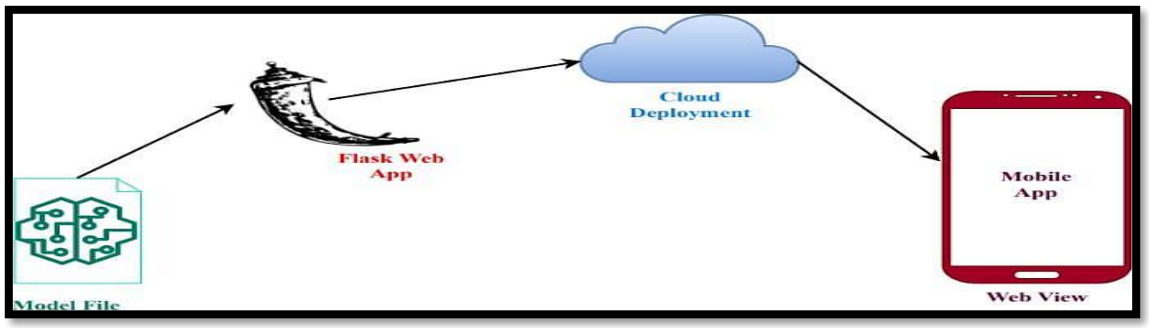


Figure 4. Backend flow of an Application

This section will give the complete overview of how project is implemented right from training to deploying the app.

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

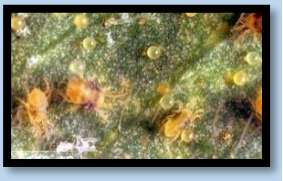
A. Training the model

In order to train the model, the dataset is prepared based on five classes, which is discussed earlier in Introduction (Section I). PlantVillage (for Tomato, Potato, Cherry & Corn) & UCI machine learning repository, created by Jitesh et. al. [28] (for Rice) were utilized, with 11,727 images in dataset. Table 2 shows the details of classes and their sample images. A good GPU in google colab, called as "Tesla T4" is selected to train the model faster and effectively. Data pre-processing involves the Data Augmentation is applied (as discussed earlier in Literature Survey) to avoid under- fitting. Adam optimizer is considered along with categori- cal cross entropy as loss function with learning rate of e^-3 in training the model. Parameters for different layers were experimentally modified depending training efficiency by applying convolution & MaxPooling (chooses the max of all values) layer and flatten the feature to 1-D. Regularization (L2 or Ridge Regression) is also applied to minimize the error and avoid overfitting. Dropout is also added and modified depending on training loss and overfitting of data. Best model (95.42% training accuracy) was saved among all obtained during training the model in 100 epochs. This can be achieved by model checkpointing [29] which is available in Tensorflow. "Fig. 5" explains the program flow of training. Saved model will be utilized by the flask webapp to read the input image and run the model to obtain result. Percentage of infection and invalid image detection will be added with the model to run simultaneously in a single program.

Table 2: Details of different classes of the dataset and their sample images.

Disease Class Name	Caused By	Sample Image
Tomato Early Blight	Alternaria Saloni (Fungal Pathogene)	
Tomato Late Blight	Phytophthora (Omycete Pathogene)	
Tomato Mosaic virus	Small wound (Virus)	
Tomato Septoria leaf spot	Septoria Lycopersici (Fungus)	
Tomato Yellow leaf curl virus	Leaf curl upward/downward (Virus)	

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

Tomato Leaf mold	Cladosporium Fubucum (Pathogen)	
Two-spotted spider termites	Kind of Termites look like spider and built web	
Tomato Bacterial spot	**Four species of Xanthomonas**	
Tomato Healthy shape	Not Applicable	
Corn Healthy	Not Applicable	
Corn Common Rust	Puccinia Sorghi (Fungus)	
Corn Cercospora Grey spot leaf	Cercospora Zeae-maydis (Fungus)	
Corn Northern Leaf Blight	Exerohilum Turcicum (Fungus)	

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

Cherry Healthy	Not Applicable	
Cherry Powdery Mildew	Podosphaera Clandestina (Fungus)	
Rice smut	Etyloma Oryzae (Fungus)	
Rice Brown spot	Cochliobolus Miyabeanus (Fungus)	
Rice Bacterial leaf Blight	Xanthomonas Oryzae (Bacteria)	
Potato early Blight	Alternaria Solani (Pathogene)	
Potato late Blight	Phytophthora Infestans (Fungus)	
Potato Healthy	Not Applicable	

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

B. Webapp Development & Deployment

Following steps were implemented to develop and deploy the webapp:

1) Load the Model: Since Flask is based on python, it is easier to implement the model and webapp at same time. We can load the saved model using same library, which is used to train the model and define all the classes used in training to display the output.

2) Reading images: All the user input images can be read using computer vision (CV2) library and can be used for predicting the class along with infection or validity.

3) Infection Percentage calculation & Validating the Image: As described earlier, Image Thresholding is used for calculating amount of infection and checking the validity of image. A logic behind the calculating the infection is simply the amount of black area within the total area of leaf, given by:

((Total Area – Green Area)/(Total Area))*100

Edge Detection will play its role in segregating the leaf from its background to consider only required area. Excess Green Indexing function will be created based on green components of Red-Green-Blue (RGB) and from this we will convert image into grayscale to obtain image histogram. Otsu's algorithm along with Iterative Thresholding Selection is applied on it to obtain the optimum threshold value to binarize the image. The output will be improved using a median blurring filter, and the remaining area will be filled in for computing the total area based on borders found using edge detection. Grayscale will now be used to detect spots or infected areas, and the green area will be calculated and compared to a copy of this image without the backdrop. We can determine the infection level in this method as shown in "Fig. 6", in which shows the infection percentage to be 86.573%.

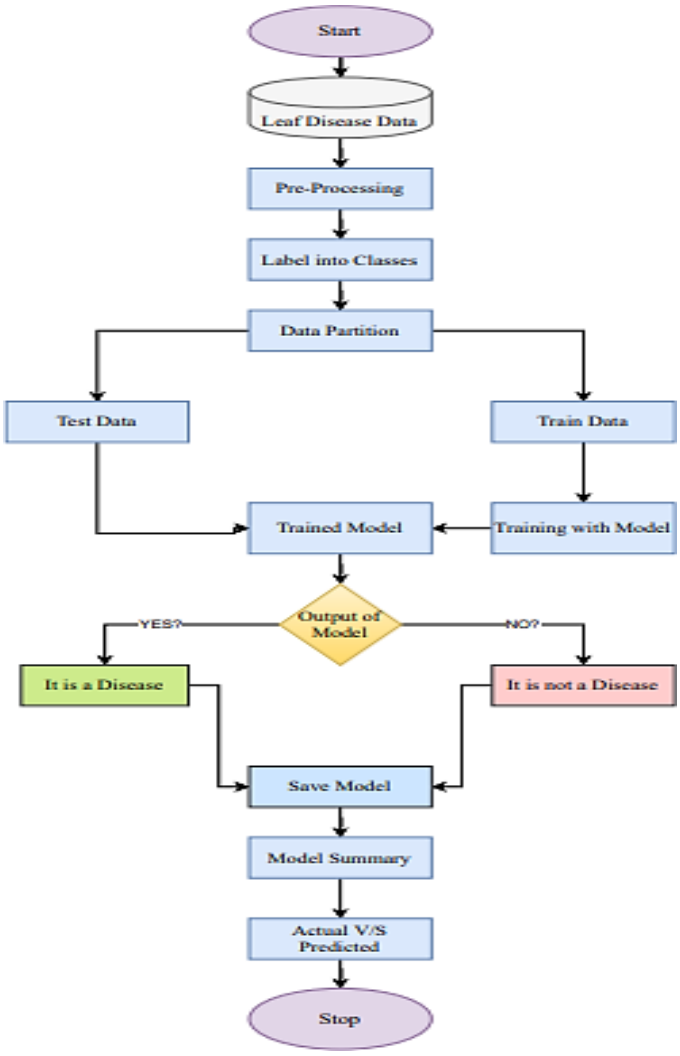


Figure 5. Program flow of training the model

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

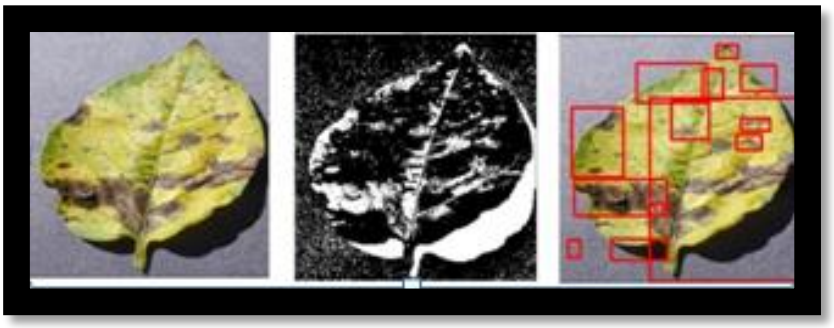


Figure 6. Calculating Percentage of Infection of a leaf

Coming to checking the validity of image, simply the threshold value is set depending on amount of green components present in image. If they are less than threshold value, it will simply return the image as invalid otherwise it will be analyzed to obtain the predicted output and amount of infection

4) Cloud Deployment: As discussed earlier, for running our project's backend or server side app, we need the cloud deployment solution of PaaS model. Among the several cloud service providers, Heroku is considered for deploying the webapp for ease of access and good availability of slug size for basic plan [30]. On successful completion of developing the interface for webapp, it will be ready to push to cloud and run the application. All the necessary com- mands required to deploy/push will be available on Heroku's Dashboard on successful sign-up & creating a plat- form to run the app.

C. Android App Development

Once the app is pushed successfully and the link is obtained to run it, we will utilize it in our android app using WebView layout. Below mentioned are the details of app development process:

1) Logo Design & Naming the App: Always apps are identified to be unique with their names & logos. A good Logo and unique name distinguishes the app to be unique and gives the good first impression and trust on its working. App is named as "PestFree" which depicts the app aims at the plant to be free of pests which in turns produces a good yield and satisfaction to the user, which shows the leaf surrounded by Tomato, Potato, Cherry, Corn & Rice signifying the app's objective of detecting diseases of leaves belonging to mentioned classes of plants and no pest in background (trans- parent) to depict its aim like name.

2) WebView Layout: App is developed in Android Studio, which runs on Java Language. It has view class consisting of several layouts to display the app design to users. Some of examples are RelativeLayout, ScrollView, LinearLayout etc., out of which WebView is one of them. We will utilize the link generated by Heroku to run the webapp in Android app's interface, which will be set depending on dimensions of phone & responsive design of the webapp.

3) Programming required files: In order to run the app in Android, it should provide the several permissions to ac- cess such as Gallery, Camera, and Microphone for the security reasons (avoiding corrupted & malicious apps from attacking on privacy) and assure the safety for users' data. These all can be updated through Manifest files. All the required layouts for pages including the WebView page for a good interface will be written in .xml files and all the required code to run the application in java files. The FileChooser for WebView is necessary to access the Gallery of phone & upload the image. On successful completion of Android app development, we are ready to test the outputs, apply the necessary changes and deploy the app to use.

4. Results and Discussions

This section shows all the experimental/testing results of an app, tested for different images of plants and others for accuracy of output, as follows:

Table 3: Predicted vs Actual output results - 1

Predicted	Actual
Cherry Powdery Mildew	Cherry Powdery Mildew
Corn Common Rust	Corn Common Rust
Potato Early Blight	Potato Early Blight

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

Table 4: Predicted vs Actual output results - 2.

Predicted	Actual
Rice Brown Spot	Rice Brown Spot
Tomato Septoria Leaf Spot	Tomato Septoria Leaf Spot
Invalid Image	Other image (Random)
Potato Healthy	Other image (Shoe flower/Hibiscus)

As we can observe, the output results almost accurate except the case in which plant image apart from the mentioned five classes is considered and other if the other image which is has greater amount of greenery (like other plants or trees or garden/park etc.) is uploaded to predict the output. The user must be aware of this in order to avoid wrong predictions & misunderstanding the analysis of in- put. This can be mentioned in the disclaimer of the app be- fore use to spread the awareness among users, making the app robust and accurate & user-friendly to work with it.

5. Conclusion

This paper concludes that any application that is created for actual usage must be reliable and error-free. For this, a strong analysis is needed, and then a good model usually needs to be built and trained to produce correct results. Our project attempts to create such an application while taking into account the target users, their usage style, the types of user input and output sensitivity. Since users check for both plant detection and the level of infection, algorithms must be created with a very low error rate to prevent users from misinterpreting and cultivating incorrect ways based on output results. A highly accurate model as well as the excellent user interface makes this app best in its segment.

Coming to the future development, since most technologies are already automated and have high internet connectivity before 5G is introduced in India, plant disease detection may be much simpler to adopt. This technology can be used in conjunction with a good application to speed up work and efficiently utilize resources. The technology being mentioned here could be a remote-controlled robot or drone that flies across a field and takes images periodically to detect. The app can use farmer to control all of these features. Fast internet will speed up the application's operation because cloud contact makes it happen much more quickly and without latency. With the farmer's permission, a drone or robot can apply pesticide to the necessary area based on infection, maximizing its use and saving a cost of investment on buying them. A language functionality can also be added so that users from all different regions can use the app comfortably. This feature specially can work with a new feature of adding Rabi & Kharif crop making the app useful even for seasonal purpose. As technology aims at making the work efficient and faster, there is a need of good application as well to utilize them and make it reachable widely to all the users who need the help of such. Application being user-friendly and accurate will make it understand easily and making lesser errors. This way not only users but also consumers who depend on the output generated by users will be benefitted.

Funding: “This research received no external funding”

Conflicts of Interest: “The authors declare no conflict of interest.”

References

[1] P. A. Nazarov, D. N. Baleev, M. I. Ivanova, L. M. Sokolova, and M. V. Karakozova, "Infectious Plant Diseases: Etiology, Current Status, Problems and Prospects in Plant Protection," *Acta Naturae*, vol. 12, no. 3, pp. 46–59, Jul.–Sep. 2020, doi: 10.32607/actanaturae.11026.

[2] V. Singh, N. Sharma, and S. Singh, "A review of imaging techniques for plant disease detection," *Artificial Intelligence in Agriculture*, vol. 4, pp. 229–242, 2020, doi: 10.1016/j.aiia.2020.10.002.

[3] S. K. Noon, M. Amjad, M. A. Qureshi, A. Mannan, and T. Awan, "An Improved Detection Method for Crop & Fruit Leaf Disease under Real-Field Conditions," *AgriEngineering*, vol. 6, no. 1, pp. 344–360, 2024, doi: 10.3390/agriengineering601002.

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

[4] W. Shafik, A. Tufail, A. Namaun, D. Silva, and M. Apong, "A Systematic Literature Review on Plant Disease Detection: Motivations, Classification Techniques, Datasets, Challenges, and Future Trends," *IEEE Access*, Jun. 2023, doi: 10.1109/ACCESS.2023.3284760.

[5] P. Sykora, P. Kamencay, R. Hudec, M. Benco, and M. Sinko, "Comparison of Feature Extraction Methods and Deep Learning Framework for Depth Map Recognition," *New Trends in Signal Processing (NTSP)*, Liptovský Mikuláš, Slovakia, pp. 1–7, 2018, doi: 10.23919/NTSP.2018.8524109.

[6] W. B. Demilie, "Plant disease detection and classification techniques: a comparative study of the performances," *J Big Data*, vol. 11, p. 5, 2024, doi: 10.1186/s40537-023-00863-9.

[7] I. T. Ahmed, C. S. Der, N. Jamil, and M. A. Mohamed, "Improve of contrast-distorted image quality assessment based on convolutional neural networks," *International Journal of Electrical and Computer Engineering*, 2019, doi: 10.11591/ijece.v9i6.pp5604-5614.

[8] W.-Y. Chang, S.-J. Wu, and J.-W. Hsu, "Investigated iterative convergences of neural network for prediction turning tool wear," *The International Journal of Advanced Manufacturing Technology*, 2020, doi: 10.1007/s00170-019-04821-9.

[9] R. Poojary and A. Pai, "Comparative Study of Model Optimization Techniques in Fine-Tuned CNN Models," *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, Ras Al Khaimah, UAE, pp. 1–4, 2019, doi: 10.1109/ICECTA48151.2019.8959681.

[10] M. E. H. Chowdhury et al., "Tomato Leaf Diseases Detection Using Deep Learning Technique," *Technology in Agriculture*, 2021, doi: 10.5772/intechopen.97319.

[11] B. Liu, Z. Ding, L. Tian, D. He, S. Li, and H. Wang, "Grape Leaf Disease Identification Using Improved Deep Convolutional Neural Networks," *Frontiers in Plant Science*, vol. 11, p. 1082, Jul. 2020, doi: 10.3389/fpls.2020.01082.

[12] A. Ramcharan et al., "A Mobile-Based Deep Learning Model for Cassava Disease Diagnosis," *Frontiers in Plant Science*, vol. 10, p. 272, 2019, doi: 10.3389/fpls.2019.00272.

[13] C. U. Kumari, S. Jeevan Prasad, and G. Mounika, "Leaf Disease Detection: Feature Extraction with K-means clustering and Classification with ANN," *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, pp. 1095–1098, 2019, doi: 10.1109/ICCMC.2019.8819750.

[14] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," *Computers and Electronics in Agriculture*, vol. 173, p. 105393, 2020, doi: 10.1016/j.compag.2020.105393.

[15] M. Francis and C. Deisy, "Disease Detection and Classification in Agricultural Plants Using Convolutional Neural Networks—A Visual Understanding," *Proceedings of the 6th International Conference on Signal Processing and Integrated Network*, 2019.

[16] J. Zhao and J. Qu, "A Detection Method for Tomato Fruit Common Physiological Diseases Based on YOLOv2," *2019 10th International Conference on Information Technology in Medicine and Education (ITME)*, Qingdao, China, pp. 559–563, 2019, doi: 10.1109/ITME.2019.00132.

[17] V. K. Shrivastava, M. K. Pradhan, S. Minz, and M. P. Thakur, "Rice plant disease classification using transfer learning of deep convolution neural network," *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 2019.

[18] M. K. Priya and S. Dhanabal, "Analyses of Nine Different Types of Diseases in Paddy with Hybrid Algorithms using Deep Learning," *International Journal of Engineering Research & Technology (IJERT)*, vol. 8, no. 8, 2020, doi: 10.17577/IJERTCONV8IS08025.

[19] M. Singh, S. Ayuub, A. Baronia, and D. Soni, "Analysis and Implementation of Disease Detection in Leafs and Fruit Using Image Processing and Machine Learning," *SN Computer Science*, vol. 4, no. 5, 2023, doi: 10.1007/s42979-023-02045-z.

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025

[20] A. M. Ali, A. Słowik, I. M. Hezam, and M. Abdel-Basset, "Sustainable smart system for vegetables plant disease detection: Four vegetable case studies," *Computers and Electronics in Agriculture*, vol. 227, p. 109672, 2024, doi: 10.1016/j.compag.2024.109672.

[21] S. Abisha, A. M. Mutawa, M. Murugappan, and S. Krishnan, "Brinjal leaf diseases detection based on discrete Shearlet transform and Deep Convolutional Neural Network," *PLOS One*, 2023, doi: 10.1371/journal.pone.0284021.

[22] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *J Big Data*, vol. 6, p. 60, 2019, doi: 10.1186/s40537-019-0197-0.

[23] R. K. Dash et al., "Fine-tuned support vector regression model for stock predictions," *Neural Comput & Applic*, vol. 35, pp. 23295–23309, 2023, doi: 10.1007/s00521-021-05842-w.

[24] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan. 1979, doi: 10.1109/TSMC.1979.4310076.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017, doi: 10.1145/3065386.

[27] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[28] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *International Conference on Machine Learning (ICML)*, 2019.

[29] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[30] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

DOI: https://doi.org/10.54216/FPA.190218
Received: January 05, 2025 Revised: February 10, 2025 Accepted: March 03, 2025