An Automated System to Detect Plant Disease using Deep Learning

Gotlur Karuna¹, Chalamalla Eshwar¹, Yamsani Sai Bharath¹, Daliyet Rohan¹, Harish Dutt Sharma², S. M. Naveen Raja³

¹Department of CSE (AI & ML), GRIET, Hyderabad, Telangana State, India
²Uttaranchal Institute of Technology, Uttaranchal University, Dehradun, 248007, India
³KG Reddy College of Engineering & Technology, Hyderabad, India

Abstract: Crop diseases, particularly in places with weak infrastructure, represent a severe danger to the security of the global food supply. To address this challenge, a platform for accurate identification of plant diseases is needed. The paper proposes deep learning techniques to identify plant diseases. The platform uses the “CNN” algorithm, widely known for its high accuracy in image classification, enabling it to accurately identify plant diseases from images. Furthermore, the platform offers automated suggestions for preventing and provide supplements for that disease and managing the spread of crop diseases using its user-friendly web interface.

1 Introduction

Agriculture contributes a vital part in the economic advancement of countries worldwide. Among the key challenges in agriculture is the detection of plant diseases, which may result to significant harvest yield losses and economic losses for farmers. In conventional techniques Plants must be manually inspected, which is tedious, costly, and vulnerable to error, in order to identify plant diseases. There has been an increase in interest in utilising deep learning methods to automatically detect plant diseases, which has the potential to enhance the effectiveness and precision of identifying plant diseases. An automated system design for detection and management of plant diseases have been a significant concern for farmers and scientists for centuries. The earliest known record of plant disease dates back to ancient Egypt, where farmers used natural remedies to protect their crops. In the early 19th century, the Irish potato famine led to the death of over a million people due to potato blight, emphasizing the importance of plant disease detection and management.

With the advancement of technology, plant disease detection methods have evolved from traditional techniques like visual inspection and chemical analysis to more sophisticated methods like deep learning and machine learning. In the mid-20th century, scientists began using decision trees and other machine-learning techniques to categorize plant diseases based on indications. In the 1990s, researchers began exploring the use of neural networks, which is a form of machine learning algorithm capable of learning to identify similarities in data. In the early 2000s, the development of deep learning techniques revolutionized the area of

*Corresponding author: karuna.g@griet.ac.in
image recognition, and scientists began exploring deep learning and being used to detect plant diseases. Researchers from the University of Bonn in Germany proposed one of the earliest deep learning-based methods for detecting plant diseases in 2016. Convolutional neural networks (CNNs) were used to create a deep learning model that could distinguish between photos of healthy and diseased leaves. Their model's accuracy of over 99% shows the potential of deep learning methods for identifying plant diseases. Since then, many researchers have created deep-learning models for the identification of plant diseases on a variety of crops, such as tomatoes, potatoes, grapes, and apples. In order to categorise tomato leaf illnesses in 2019, researchers from the Indian Institute of Technology Roorkee created a deep learning model that makes use of CNNs and transfer learning. Their model's accuracy of 98.3% in identifying plant diseases in particular crops shows the potential of deep learning approaches.

Acquiring labelled training data for CNN implementation is time-consuming and labour-intensive, requiring excellent data quality. Overfitting is a typical problem in which CNNs specialise too much on training data, resulting in poor performance on fresh data. CNN architecture design, comprising layers, filters, and hyperparameters, is critical yet difficult. CNNs need a lot of computing power, which limits their use in resource-constrained scenarios. Furthermore, CNNs lack interpretability, making them unsuitable for use in explanation-driven applications. Domain-specific issues, like as data privacy and ethical concerns, impede CNN development in certain disciplines.

2 Literature survey

Convolutional Neural Networks (CNNs) have gained prominence in a variety of sectors due to their ability to extract meaningful features from complicated input. Image classification is a key use of CNNs, where they can recognize certain patterns or objects in photos and categorize them into distinct categories. Furthermore, CNNs excel in object recognition tasks by identifying and localizing items in real time, making them useful in fields like self-driving automobiles and security systems. CNNs are also used in face recognition technology, where their ability to learn and detect facial traits effectively allows for the trustworthy identification of individuals. Furthermore, in the realm of medical diagnostics, CNNs have demonstrated their usefulness in recognizing malignant cells in medical pictures and diagnosing illnesses from complicated medical data.

Table 1. Existing approaches.

<table>
<thead>
<tr>
<th>Ref. No.</th>
<th>Paper Name</th>
<th>Methodology</th>
<th>Results/Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>[2]</td>
<td>Plant Leaf Disease Classification using EfficientNet deep learning model</td>
<td>EfficientNet Deep Learning CNN architecture.</td>
<td>Among various deep learning models, the EfficientNet structure had the highest accuracy.</td>
</tr>
<tr>
<td>No.</td>
<td>Title</td>
<td>Model(s)</td>
<td>Details</td>
</tr>
<tr>
<td>-----</td>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>[5]</td>
<td>Tomato Crop Disease Classification using pre-trained deep learning algorithm</td>
<td>AlexNet And VGG16 CNN architectures.</td>
<td>Achieved an accuracy of 97.29% for VGG16 and 97.49% for AlexNet.</td>
</tr>
<tr>
<td>[6]</td>
<td>Plant Disease Classification using deep learning</td>
<td>CNN, RESNET, VGG, and DenseNet models.</td>
<td>Achieved an accuracy of 94.58% for CNN, 95.32% for VGG,97.04% for RESNET, and 98.27% for DENSENET.</td>
</tr>
<tr>
<td>[7]</td>
<td>Bell Pepper Leaf Disease Classification</td>
<td>CNN model.</td>
<td>Achieved an accuracy of 96.78%.</td>
</tr>
<tr>
<td>[8]</td>
<td>A Comparative Analysis of Deep Learning Models Applied for Disease Classification in Bell Pepper</td>
<td>VGG, ResNet, DenseNet, and InceptionResNet.</td>
<td>'DenseNet' requires the least amount of training time and has the greatest validation accuracy of any of the models mentioned above. It achieved a 96.87% accuracy.</td>
</tr>
<tr>
<td>[9]</td>
<td>Leaf Disease Detection using Deep Learning</td>
<td>SVM and CNN model.</td>
<td>Achieved an accuracy of 80% with SVM and 97.71 % for the CNN model.</td>
</tr>
<tr>
<td>[11]</td>
<td>Potato Leaf Diseases Detection using Deep Learning</td>
<td>Extracted features from pre-trained CNN models are passed to classifiers such as SVM, KNN, and Neural networks.</td>
<td>Achieved an accuracy of 97.8%.</td>
</tr>
</tbody>
</table>

In the paper, "Potato Leaf Disease Classification Using Deep Learning Approach," the authors use VGG16 and VGG19 deep learning models, as well as data augmentation strategies, to solve the overfitting problem in diagnosing potato leaf illnesses. Their dataset contains 5100 photos classified into five illness categories, and both algorithms attain an accuracy of 91%. In the same way, the study "Plant Leaf Disease Classification Using EfficientNet Deep Learning Model" introduces the EfficientNet CNN model for plant disease classification. Using the Plant Village dataset, this model is compared to different CNN models such as AlexNet, VGG16, Inception V3, and ResNet50. The findings show that EfficientNet beats the other models in terms of accuracy.

In another work titled "Deep Learning Utilisation in Agriculture: Detection of Rice Plant Diseases Using an Improved CNN Model," the authors categorise rice plant illnesses using a fine-tuned transfer learning VGG19 CNN model. They mitigate overfitting with data augmentation strategies and obtain an outstanding accuracy of 96.08% in recognising six illness types. The research "Classification of Beans Leaf Diseases Using Fine Tuned CNN Model" also investigates the classification of bean leaf illnesses using CNN models such as MobileNetV2, EfficientB6, and NasNet. With the Adam optimizer, EfficientNetB6 obtains the greatest accuracy of 96.62%. These articles illustrate the accuracy with which deep
learning models identify and categorise plant illnesses, opening the path for better agricultural practices.

The existing methods lack a user-friendly platform to utilize the built model. In addition, the dataset employed in these methods was split into two categories: training and testing, which may raise concerns regarding overfitting. Additionally, a significant number of these methods did not utilize data augmentation techniques, which may lead to overfitting. Authors [13] highlighted the significance of ML in prediction, pattern recognition and error reduction across diverse fields, emphasizing the impact of AI in broad domain. Authors [14] highlighted the significance of ML in prediction, pattern recognition and error reduction across diverse fields, emphasizing the impact of AI in broad domain. Authors [15] suggested data mining techniques to predict disease-prevalence based on symptoms in healthcare data. The appropriate prediction helps healthcare organizations avoid drug shortages and further ensures timely treatment of patients. Image restoration is to enhance images by removing noise and restoring them to their original quality. The present approach explored various methods in both frequency and spatial domains, followed by analysing their performance using simulations.

3 Proposed method

Convolutional Neural Networks (CNN or ConvNet) are a fascinating subset of machine learning that has transformed the field of computer vision. While machine learning techniques encompass a wide range of algorithms and models, CNNs are designed for tackling image-related tasks. Their architecture and working principles make them incredibly effective at processing visual data and extracting meaningful features. Its design principles are adapting already trained model in order to automate the system completely. At the core of a CNN lies its ability to uncover intricate patterns and details within images. By leveraging principles from linear algebra, CNNs analyse the binary representation of visual data to identify specific features that are relevant for classification and recognition. Interestingly, CNNs may also classify other forms of data, such as time series, audio signals, and text.

Fig. 1. Basic architecture of convolutional neural network (Courtesy: Source [13]).
3.1 Objectives

Plant diseases present a huge danger to food security and the agriculture business generally. When a plant gets infected, it can cause a reduction in crop yield, ultimately leading to a shortage of food. These effects are particularly devastating for developing countries, where many people depend on agriculture for their livelihoods. Thus, it is crucial to detect plant diseases accurately and quickly to stop the spread of the illness to other plants. Traditionally, the investigation and diagnosis of plant diseases are relied on visual inspection by trained experts. However, this approach is time-consuming and labour-intensive, making it difficult to scale up and deploy in large agricultural areas. The use of machine learning algorithms is another method for identification. Despite the availability of numerous classification models for detecting plant diseases, there are still limited online platforms available for utilizing these models to identify specific diseases. This lack of infrastructure can make farming more challenging and other stakeholders to access accurate information and make informed decisions about disease management.

- Create a strong CNN model capable of reliably classifying pictures. The model will be developed with images of various illnesses from a dataset. The CNN architecture is designed to extract meaningful information from images and forecast based on those attributes.
- CNN model will be specifically designed to classify images and detect diseases. It will be trained on a diverse dataset that includes images representing various diseases. The objective is to achieve an automated design to achieve high accuracy in classifying and identifying the presence of a particular disease in an image.
- Alongside the CNN model development, the paper aims to create a user-friendly website for deploying the model. The website will provide an intuitive interface for users to upload their images for disease detection. It will have clear instructions on how to capture and upload images and ensure a smooth user experience.
- The website will integrate the trained CNN model to analyse the uploaded images and detect the presence of diseases. Based on the detection results, the system will provide recommendations for disease prevention and management. These recommendations may include treatment options, preventive measures, which is automated.

3.2 Architecture

To use deep learning techniques to identify plant diseases. The platform uses the "CNN" algorithm, widely known for its high accuracy in image classification, enabling it to accurately identify plant diseases from images. Furthermore, the platform offers recommendations for preventing and managing the spread of crop diseases using its user-friendly web interface. Figure 2 represents the architecture diagram of building the CNN model. It shows the steps involved in building the CNN model. It also represents the flow in which data travels while building the model.

1. First, the dataset is pre-processed. The pre-processing includes a wide range of tasks like resizing images, converting images to tensors, and augmenting data using rotation, flipping, or zooming of images.
2. The pre-processed data is then split into 3 datasets- train, test, and validation data in the ratios 8:1:1 respectively.
3. The model is built on train data and validation data acts as the first test data which is used for overcoming the overfitting problem.
4. The model built is then evaluated using the test data and the evaluation metrics such as accuracy and loss are noted and the model is saved.
5. The model showing the best results is integrated with the web interface which utilizes this model to classify images.

**Fig. 2.** Flow diagram of building CNN model.

**Fig. 3.** Flow diagram of User-Interface.
3.3 Steps in proposed method

In this paper, transfer learning is used as it takes too much time to build and train the models. Transfer learning is used by leveraging the pre-trained models VGG16, VGG19, InceptionV1, and DenseNet201. The models are fine-tuned by removing the last layer of the pre-trained model and adding new layers to improve their performance. The weights employed are based on the ImageNet dataset, which contains 14 million photos from 1000 distinct classifications.

- **Step-1**: Data Collection: The dataset used is from the Kaggle repository named “PlantVillage.” The dataset consists of 21,000 images of 3 different species named Tomato, Potato, and Bell Pepper and diseases of 15 categories.

- **Step-2**: Data Pre-processing: Preparing images for use in a Convolutional Neural Network (CNN) architecture is referred to as data pre-processing. The first step is to resize the images so that they fit into the architecture's input layer.

- **Step-3**: Data Splitting: Data splitting refers to dividing the dataset into different subsets for the development of the model. The dataset used here is split into three subsets—training data, testing data, and validation data of divisions of 80-10-10. The training data is data used for the development of the model.

- **Step-4**: Model Development: Model Development refers to the process of developing the model by designing its architecture, and selecting the optimizers to be used. Designing the model architecture is a critical stage in model development, since its automated design process.

  For VGG19, firstly, the input layer is added to the model, followed by two sequential layers, the first layer for resizing and rescaling the images and the second layer for augmenting images. Later, the pre-trained VGG19 network is added followed by an average pooling layer, a dense layer, a normalization layer, a dropout layer, and finally output layer (i.e., a dense layer) is added to the model. Fig 4 depicts the architecture of the model developed.

![Fig. 4. Architecture of model built by using VGG19 network.](image)

For DenseNet201, firstly, the input layer is added to the model, followed by two sequential layers, the first layer for resizing and rescaling the images and the second layer for augmenting images. Later, the DenseNet201 network is added to the model the average pooling layer is followed by a dense layer, a batch normalization layer, and a dropout layer., and finally the output layer (i.e., a dense layer) is added to the model. Fig. 5 symbolises the architecture of the model developed.

- **Step-5**: Model Training: The process of refining a model created with the aid of training data is known as model training. The model training process includes selecting the hyperparameters like batch size, and epochs.
The epoch value taken while training the model is 50 and the batch size was 32. The models are trained using the Adam optimizer, which is known for showing the state-of-the-art results.

Fig. 5. Architecture of model built by using DenseNet201 network.

Fig. 6. Comparison of trainable parameters.

Fig. 7. Comparison of non-trainable parameters.

Figures 6 and 7 show the comparison between the total number of trainable and untrainable characteristics of the models used. Using the number of parameters, we can compare the computation time and complexity of the models. The model with more trainable parameters takes more time to train the model.
Step-6: Model Evaluation: Model assessment is the step in which the created model is assessed for its performance on predictions using metrics such as accuracy and loss. The model assessment step evaluates the model developed using testing data. The model with the highest accuracy is chosen. Accuracy is a popular assessment statistic used to assess a model's performance. It is defined as the percentage of right predictions out of all forecasts made. 
\[ \text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \]
TP - number of true positives, TN - number of true negatives, FP - number of false positives, FN - number of false negatives.
Here, the model's evaluation is done based on two evaluation metrics accuracy and loss. The accuracy and loss are calculated for all three datasets - train, test, and validation.

Step-7: User Interface: User Interface phase is the process of developing a user interface for integrating the model to use the model easily. The user interface here is developed using HTML, CSS, and frameworks like Bootstrap.

The Fig 8 shows the index page of the user interface. This page contains information about the application, how it works, and what type of plant diseases it can detect. When the “AI Engine” button is selected a new page opens where the user may submit a picture plant to be detected as shown in the fig.9.

Step-8: Model Deployment: Model Deployment phase is the crucial phase, where the developed model is integrated with a web user interface using Flask API along with the information related to the diseases like preventive measures and management of specific diseases.

The supplements_info dataset contains all the details about supplements and where to find them and the disease_info dataset contains the details about the preventive measures of the diseases and also the description of the diseases. When the model and datasets are integrated with the user interface, the interface can display the Supplements page by using the supplements_info dataset as shown in Fig 10 and the final results are displayed as shown in Figure 11.
When the submit button is clicked, the picture has been pre-processed and sent as the input to the model and the results are displayed in Fig 3.23. The model is then updated by the input and the updated model is sent to Flask API and it replaces the updated model with the previous model. The description of the disease and preventive measures are taken from the disease_info based on the classification of the image and also the details about supplements are taken from the supplements_info dataset using the Django framework and are displayed as shown in fig 11. When the supplements button is clicked the user interface leverages the supplements_info dataset using Django and displays results as shown in fig 10.

4 Results and discussion

The dataset used for building model is named as the Plant Village. It is sourced from the Kaggle Repository. It encompasses a diverse collection of 20,680 images, all uniformly sized at 256 x 256 pixels. These images are categorized into three plant species: Potato, Tomato, and Bell Pepper. The primary focus of this dataset revolves around the study of plant diseases, with images capturing various afflictions across 15 distinct categories.

Apart from the Plant Village dataset, two datasets named supplements_info.csv and disease_info.csv are also used. The Supplements information dataset contains information about the supplements and the links to purchase the supplements. For the purpose of identifying plant diseases, four deep-learning convolutional neural networks (CNN) algorithms are employed. These algorithms include VGG16, VGG19, GoogleNet (InceptionV1), and DenseNet201. The performance of these models is calculated using accuracy and loss metrics to determine the best-performing algorithm. The evaluation process involved monitoring the accuracy of each algorithm throughout the training epochs. Accuracy refers to the model's ability to classify plant disease images accurately. By observing how the accuracy of each algorithm evolves with each epoch value, important insights can be gained about their learning capabilities and performance trends. The following graphs display the changes in accuracy for each algorithm across different epochs. These visualizations provide a comprehensive understanding of how the models' accuracy improves or fluctuates during the training. Analyzing these graphs will help identify patterns, trends, and potential areas for optimization in each algorithm's learning process.

This graph represents the link between the number of epochs and a model's accuracy on a dataset during the training and testing procedure. The x-axis indicates the number of epochs, which refers to the complete passes made by the model through the training data. The Accuracy is represented on the y-axis and measures how well the model narrows the gap between the targets it predicts and the actual outputs.
### Table 2. Data description

<table>
<thead>
<tr>
<th>Disease</th>
<th>Species</th>
<th>Number of Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bacterial Spot</td>
<td>Tomato</td>
<td>2127</td>
</tr>
<tr>
<td>Early Blight</td>
<td>Tomato</td>
<td>1000</td>
</tr>
<tr>
<td>Late Blight</td>
<td>Tomato</td>
<td>1909</td>
</tr>
<tr>
<td>Leaf Mold</td>
<td>Tomato</td>
<td>952</td>
</tr>
<tr>
<td>Septoria Leaf Spot</td>
<td>Tomato</td>
<td>1771</td>
</tr>
<tr>
<td>Spider Mites</td>
<td>Tomato</td>
<td>1676</td>
</tr>
<tr>
<td>Target Spot</td>
<td>Tomato</td>
<td>1404</td>
</tr>
<tr>
<td>Yellow Leaf Curl Virus</td>
<td>Tomato</td>
<td>3209</td>
</tr>
<tr>
<td>Mosaic Virus</td>
<td>Tomato</td>
<td>373</td>
</tr>
<tr>
<td>Healthy</td>
<td>Tomato</td>
<td>1591</td>
</tr>
<tr>
<td>Bacterial Spot</td>
<td>Bell Pepper</td>
<td>997</td>
</tr>
<tr>
<td>Healthy</td>
<td>Bell Pepper</td>
<td>1478</td>
</tr>
<tr>
<td>Early Blight</td>
<td>Potato</td>
<td>1000</td>
</tr>
<tr>
<td>Late Blight</td>
<td>Potato</td>
<td>1000</td>
</tr>
<tr>
<td>Healthy</td>
<td>Potato</td>
<td>152</td>
</tr>
</tbody>
</table>

**Fig. 12.** Some Images from plant village dataset.
The relationship between the loss and the number of epochs used to train and test a model on a dataset is shown in this graph. The epochs on the x-axis correspond to the total number of training passes that were successfully completed while training. The loss, which is a measurement of how well the model is doing in terms of its capacity to minimise the discrepancy between the projected outputs and the actual targets, is represented on the y-axis. In this particular instance, the model's training and testing were carried out using a learning rate of 0.1. A hyperparameter called learning rate controls how big of a step the optimisation method takes during each iteration. In order to reduce the discrepancy between expected outputs and actual targets, it affects the rate at which the model updates its parameters.

Each of the 50 epochs in the training method represented a full pass through the training data. The model does two passes during each epoch: a forward pass during which the input data travels through the neural network; and a backward pass during which the model modifies its parameters in order to optimise the loss function. Multiple epochs are intended to help the model to gradually learn from the data and improve its predictions over time.

Table 3. Evaluan metrics of the models used.

<table>
<thead>
<tr>
<th>Model</th>
<th>Optimizer</th>
<th>Accuracy</th>
<th>Loss</th>
<th>Validation Accuracy</th>
<th>Validation Loss</th>
<th>Test Accuracy</th>
<th>Test Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td>Adam</td>
<td>93.86%</td>
<td>26.01%</td>
<td>90.82%</td>
<td>27.97%</td>
<td>90.86%</td>
<td>27.54%</td>
</tr>
<tr>
<td>Google Net</td>
<td>Adam</td>
<td>99.18%</td>
<td>2.52%</td>
<td>94.58</td>
<td>24.47%</td>
<td>94.32%</td>
<td>26.57%</td>
</tr>
<tr>
<td>VGG19</td>
<td>Adam</td>
<td>92.53%</td>
<td>21.84%</td>
<td>89.60%</td>
<td>34.16%</td>
<td>36.49%</td>
<td>89.91%</td>
</tr>
<tr>
<td>DENSENET201</td>
<td>Adam</td>
<td>99.35%</td>
<td>2.22%</td>
<td>99.17%</td>
<td>6.28%</td>
<td>98.82%</td>
<td>6.93%</td>
</tr>
</tbody>
</table>
The table above gives specific information on the assessment criteria utilized, such as the accuracy and loss of the models. By comparing the above results, it is clear that DenseNett201 has shown the maximum accuracy for all types of data. It has achieved an accuracy of 99.17% for the validation dataset and 98.82% for the test dataset.

5 Conclusion and future scope

A major global worry is crop failure and how it affects the availability of food. Numerous obstacles face farmers today, such as the usage of harsh chemicals and climate change, both of which reduce yields. Deep learning in particular, a development in technology, offers encouraging methods to lessen these issues. In the suggested study, authors created a deep learning model with an astonishing accuracy of 98.82% for identifying various plant diseases. Although this accomplishment is admirable, there are still opportunities for advancement and innovation. It is critical to enlarge the dataset used for training in order to improve the model's accuracy and real-world applicability. Including a diverse range of real-world images, encompassing different lighting conditions, backgrounds, and plant growth stages, would provide the model with a more comprehensive understanding of disease patterns. This expansion would help the model generalize better and improve its performance when exposed to various environmental conditions.

Additionally, integrating this technology into a surveillance system would further automate the disease detection process. By employing computer vision and deep learning algorithms, the system could continuously monitor large agricultural areas, detecting diseases in real-time. This automated system design leads to a proactive approach would enable farmers to respond swiftly, implementing precise interventions help reduce the spread of diseases and reduce crop losses.

References

7. M. Bhagat, D. Kumar, R. Mahmood, B. Pati, M. Kumar, Bell Pepper Leaf Disease Classification Using CNN, in Proceedings of the 2nd International Conference on Data, Engineering and Applications (IDEA), Bhopal, India (2020)


