P2: Plant Disease Detection System for Sustainable Agriculture

A Project Report

submitted in partial fulfillment of the requirements

of

AICTE Internship on AI: Transformative Learning
with
TechSaksham – A joint CSR initiative of Microsoft & SAP

by

Sadneya Samant, sadneyasam05@gmail.com

Under the Guidance of

P. Raja, Master Trainer, Edunet Foundation

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who have supported and contributed to the completion of this thesis.

First and foremost, I extend my deepest thanks to my distinguished supervisor, P. Raja Sir (Master Trainer), Jay Rathod Sir and Pavan Sumohana Sir, whose exceptional guidance and mentorship have been fundamental to the success of this research. Their insightful feedback, thoughtful suggestions, and constructive critiques have not only motivated me but have also significantly shaped the direction and outcome of this project. The confidence they placed in me and their continuous support have been invaluable throughout this journey.

I would also like to extend my heartfelt thanks to the Edunet Foundation and TechSakhsham, as well as the entire team, for providing this virtual training internship free of charge. The experience gained through their support in the field of AI and Machine Learning has been invaluable and will undoubtedly contribute to my future endeavours.

Thank you once again for your unwavering support.

ABSTRACT

Plant diseases pose a significant threat to agricultural productivity, leading to substantial crop losses globally. Accurate and timely detection is crucial for effective disease management and ensuring food security. In this project, we propose an automated solution using deep learning techniques to detect plant leaf diseases. We leverage Convolutional Neural Networks (CNNs) implemented through TensorFlow to build a robust disease detection model. High-resolution images of plant leaves are analysed, enabling accurate classification of various diseases affecting crop plants. TensorFlow facilitates efficient training and optimization of the CNN model, ensuring high accuracy and computational efficiency.

To provide a user-friendly interface and seamless integration with existing systems, we developed a web application using Streamlit. This application serves farmers and stakeholders in the Indian agriculture sector, allowing easy access to the disease detection system. By identifying diseased plants early, farmers can implement timely preventive measures, such as targeted treatments and improved crop management, reducing losses and enhancing productivity.

The project aims to revolutionize disease management in Indian agriculture and empower farmers with a valuable tool to optimize crop yield. The integration of deep learning, TensorFlow, and Streamlit demonstrates the potential of combining advanced technologies to address real-world challenges in agriculture. Thus this application will help them to detect plant disease and

Keywords: Plant leaf disease detection, deep learning, convolutional neural networks, TensorFlow, Streamlit, agriculture, crop management, farmer assistance, India.

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Problem Statement:

Today presence of plant diseases significantly affects agricultural productivity, thus leads to crop losses and reducing yields. Use of traditional methods make it more time-consuming and also requires more expertise. To address this issue there is a need of automated plant disease detection system that can accurately identify plant diseases using computer vision and machine learning. Thus, such system will help farmers to make decisions based upon the disease identification.

1.2 Motivation:

Pests, diseases adversely affects the crops and the plants which has become a challenge to the farmers. In order not to lose the crops, and protect and treat the crops timely, it is important that the detection is done timely , well in advance so that appropriate actions are taken. Thus, importance and the immediate need of this model in the field of agriculture is the timely and the accurate detection of the plant disease. This research is basically a try to improve the accuracy and precision of the detection of plant diseases by using the deep learning techniques like CNN.

The study is in line with precision agriculture's guiding principles, which emphasize using technology to enhance agriculture methods. The system attempts to deliver accurate and focused and precised insights regarding plant health through the integration of deep learning models. This not only helps with early disease diagnosis but also makes it easier for farmers to make better decisions, which enables resource-efficient and sustainable farming methods.

This initiative offers a cutting-edge technology solution to lessen the impact of plant diseases, which tackles a vital issue of global food security. The development of robust disease detection systems becomes increasingly important as the globe struggles to meet the nutritional needs of an expanding population. The goal of this initiative is to significantly contribute to the security of the world's food supply.

To sum up, the project is important because it has the possibility to transform plant disease detection through the help of deep learning techniques. Through the integration of CNN and ensemble stacking, the study aims to improve disease.

1.3 Objective:

The main objective is to design and develop a machine learning model plant disease detection system that automatically identifies diseases occurring in plants. That is it checks whether the given input image shows that the particular plant is diseased or not.

1. To collect a dataset of plant leaves, including both healthy and diseased types, covering various plant species and environmental conditions to ensure diversity.
2. To Preprocess collected dataset by using pooling, flattening, dense and dropout the dataset to improve model performance and reduce overfitting.
3. To design and implement deep learning models with a focus on convolutional neural networks (CNNs) for feature extraction and transfer learning approaches using pre-trained models.
4. To Analyse model performance of deep learning model using metrics. Then, visualize and interpret model predictions to understand distinguishing features of diseased and healthy leaves.
5. To provide insights for future integration into user-friendly platforms, enabling real-time detection and recommendations for disease management in agricultural practices.

1.4 Scope of the Project:

The scope of this plant disease detection project is to develop a system capable of accurately identifying plant diseases using deep learning techniques. The project will focus on the following key areas

1. **Disease Detection:**
   The primary objective of this project is to create a system that can detect diseases in plants, particularly focusing on leaves. Using image processing and machine learning, the system will analyse leaf images and classify them into categories, such as healthy or diseased, identifying the specific disease if present.

2. **Machine Learning Model:**

   The system will employ a Convolutional Neural Network (CNN) for the image classification task. The scope includes the training, testing, and evaluation of the CNN model to ensure high accuracy in disease detection.

3. **Image Preprocessing:**

   The project includes preprocessing plant leaf images by resizing, denoising, and normalizing them to improve the performance of the disease detection model.

4. **User Interface (UI):**

   A simple and user-friendly interface will be developed for users to upload images of plant leaves. The frontend will display the results of disease detection, providing the user with actionable recommendations.

5. **Disease Classification:**

   The system will classify diseases based on existing datasets, with the possibility of expanding the dataset for more plant types and diseases. The model will classify diseases by learning from labeled images in the dataset.

6. **Real-Time Application:**

   The project aims to provide a real-time disease detection system. The user will upload images, and the system will process the image and output the diagnosis in a matter of seconds.

CHAPTER 2

Literature Survey

2.1 Review of Relevant Literature

Research on plant disease detection has shown significant advancements, particularly with the use of machine learning and deep learning techniques. Some key studies include:

1. **Verma, Gaurav et al. (2019):**
    - Developed a CNN-based system to detect and classify diseases in rice crops.
    - The model achieved accurate results by analysing images of healthy and diseased leaves.
2. **Shah, Nikhil and Jain, Sarika (2019):**
    - Used an ANN to detect diseases in cotton leaves.
    - Focused on extracting features from leaf images, resulting in reliable disease detection.
3. **Kumari, Ch. Usha (2019):**
    - Introduced a two-step approach involving K-means clustering for extracting important features and ANN for classification.
    - This method improved the efficiency and accuracy of disease detection.
4. **S. D.M., Akhilesh et al. (2019):**
    - Studied bacterial blight detection in pomegranate plants using image-based techniques.
    - Their method used image processing and machine learning to classify healthy and diseased plants.
5. **Al-Hiary, H. et al. (2011):**
    - Proposed a fast and accurate system to identify plant diseases using image processing and feature extraction.
    - Their study demonstrated how machine learning could diagnose diseases across different plants.

6. **Kumar, M. et al. (2020):**
    - Focused on detecting coffee leaf diseases using CNNs.
    - Their model showed high accuracy, emphasizing CNNs' effectiveness in agricultural applications.

2.2 Existing Models, Techniques, and Methodologies

Several techniques have been developed for plant disease detection:

- **CNNs (Convolutional Neural Networks):**
  Convolutional Neural Networks (CNNs) are a class of deep learning models that have proven to be highly effective in analyzing and classifying images. Due to their ability to automatically extract features like edges, textures, and patterns from images, CNNs excel in the task of plant disease detection. CNNs are especially powerful for classifying diseases from leaf images, where subtle differences in texture or color can signify the presence of a specific disease. They have been widely adopted for tasks such as disease classification, disease severity estimation, and even the detection of pests on plant leaves.
- **ANNs (Artificial Neural Networks):**
  Artificial Neural Networks (ANNs) are another widely used technique in plant disease detection. These models mimic the human brain's neural network structure and are often used for simpler classification tasks. ANNs are typically trained on input data that has undergone feature extraction using methods like K-means clustering or principal component analysis (PCA).Image Processing:
  Methods like texture analysis, edge detection, and segmentation help preprocess images for better disease identification.
- **Image Processing Techniques**: Traditional image processing methods are frequently used as preprocessing steps before applying machine learning models. Techniques such as **texture analysis**, **edge detection**, and **segmentation** are essential for extracting meaningful features from plant images. These methods help identify key characteristics such as the shape, color, and texture of the leaves, which are often indicative of specific diseases. **Edge detection** algorithms, such as

the **Canny edge detector**, highlight the boundaries of different regions within an image, making it easier to identify the affected areas. **Texture analysis** involves analyzing the surface characteristics of the leaves, and **segmentation** helps divide the image into meaningful regions for further analysis. By applying these image processing techniques, the data fed into machine learning models becomes more structured and informative, leading to better prediction results.

- **Transfer Learning:**

  Transfer learning involves utilizing pretrained models to improve the performance of a new model, especially when there is limited data. In the context of plant disease detection, pretrained models like **MobileNet**, **ResNet**, and **VGGNet** are used to recognize general patterns in images, which can then be fine-tuned for specific plant diseases. This approach significantly reduces the training time needed, as the model already has learned important features from a large dataset.

- **IoT Integration:**

  The integration of **Internet of Things (IoT)** technology into plant disease detection systems has become an emerging trend, enabling real-time monitoring and early detection of diseases. IoT systems consist of interconnected sensors that collect data from the field, such as environmental conditions (temperature, humidity, soil moisture, etc.) and plant health data (e.g., leaf color, shape, and texture). This data is then transmitted to cloud platforms for further analysis, often using machine learning models like CNNs. By incorporating IoT technology, farmers can monitor their crops continuously and receive immediate alerts if a disease is detected. This integration helps in making timely decisions regarding treatment, thus minimizing the spread of diseases and ensuring healthier crops. The real-time aspect of IoT-based systems can significantly enhance the effectiveness of plant disease detection, especially in large agricultural settings.

2.3 Gaps and Limitations in Existing Solutions

Despite significant advancements in plant disease detection, several challenges persist, hindering the widespread adoption and effectiveness of current systems.

1. **Limited Generalization**: Many machine learning models perform well when trained on specific crops or diseases but struggle to generalize across diverse datasets. This lack of versatility limits their applicability to a wide range of crops or new diseases that may emerge, making it difficult for farmers to rely on these models in varied agricultural environments.
2. **Environmental Variability**: Plant disease detection models often face difficulties under different environmental conditions, such as varying lighting, complex backgrounds, or overlapping symptoms of multiple diseases. These factors can significantly impact the model's accuracy and reliability, especially when capturing images from the field is challenging due to inconsistent weather or changing lighting conditions.
3. **Real-Time Use**: While many plant disease detection systems exist, few are fast enough or optimized for real-time applications in the field. Real-time disease detection is essential for timely intervention, but current systems are often too slow or computationally expensive to provide farmers with immediate feedback when they need it most.
4. **Farmer Accessibility**: A significant barrier to adoption is the lack of accessibility for farmers, especially those with limited technical knowledge. Many disease detection systems are designed with advanced users in mind or require specialized hardware, making them unsuitable for the average farmer. Additionally, not all systems are mobile-friendly, limiting their use in the field where farmers may not have access to powerful computing resources.

Now with the help this project it aims to overcome the challenges outlined above by implementing solutions that address each gap with advanced technologies and a focus on usability.

1. **Comprehensive Dataset**: To improve the model's generalization capability, this project incorporates a diverse and extensive dataset that includes images of

multiple crops and diseases. This broader dataset will help the model recognize and diagnose a wider range of plant diseases, making it more versatile and effective in real-world applications across different agricultural settings.

2. **Real-Time Detection**: One of the core objectives of this project is to develop a lightweight model optimized for real-time disease detection. By using efficient algorithms and optimizing the model for mobile devices or IoT systems, this project ensures that farmers can receive immediate feedback on the health of their crops, enabling quick intervention to prevent the spread of diseases.
3. **Enhanced Accuracy**: The project will use advanced CNN models and ensemble techniques to improve the accuracy of predictions, even in challenging scenarios. These techniques will help address issues such as overlapping symptoms or variable environmental conditions, which often lead to inaccurate predictions in existing systems. This will ensure that the model is robust and reliable under a wide range of conditions.
4. **User-Friendly Design**: Recognizing the importance of accessibility, this project focuses on creating a user-friendly design for the application. By ensuring that the system is simple, intuitive, and easy to use, even for farmers with limited technical expertise, the project aims to empower them to monitor and manage their crops effectively. The application will be optimized for mobile devices, allowing farmers to use it in the field with ease.

CHAPTER 3

Proposed Methodology

3.1 System Design

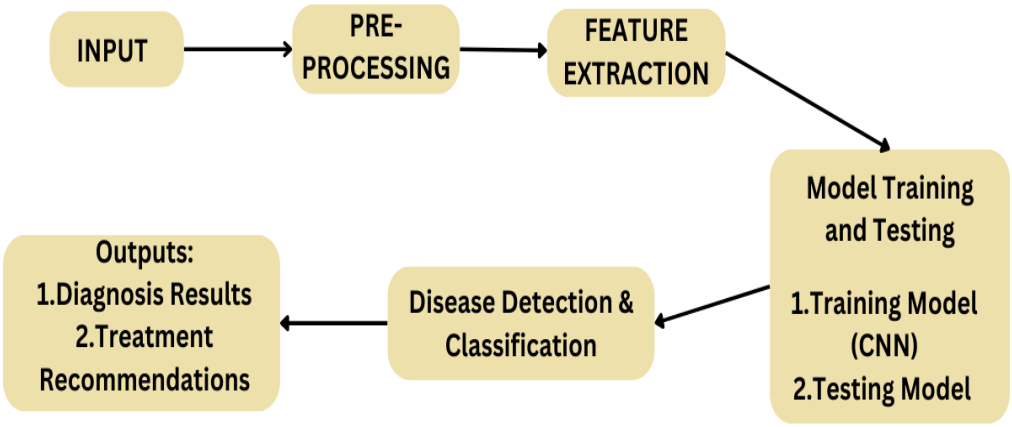

Fig1. System Architecture

1. Data Input:

The system begins by collecting images of plant leaves, either through direct upload by users or real-time capture via a connected camera. These images serve as the foundation for disease detection. To ensure compatibility with the system, users can upload images in common formats like JPEG or PNG. The system is designed to handle various input resolutions and lighting conditions.

2. Preprocessing:

Before analysis, the input images undergo preprocessing to improve their quality and standardize them for the model.

- **Resizing:** Images are resized to a uniform dimension required by the model

3. Feature Extraction:

Once preprocessed, the images are analysed to extract relevant features.

- **Texture Features:** Analysis of leaf texture patterns helps identify diseases characterized by unique surface anomalies like spots, streaks, or patches.
- **Color Features:** Variations in color (e.g., yellowing, browning, or discoloration) are key indicators of plant health.
- **Shape Features:** Structural deformations in the leaf outline or size are also considered.

  Feature extraction ensures that only the most relevant data is passed to the model, reducing computational overhead while retaining accuracy.

4. Model Training and Testing:

- A **Convolutional Neural Network (CNN)** is used as the primary model for disease detection. CNNs are well-suited for image classification due to their ability to learn spatial hierarchies of features.
- The dataset includes labeled images of healthy and diseased plant leaves, categorized based on common plant diseases.
- The training process involves feeding labeled data into the model, allowing it to learn and generalize patterns associated with each disease category.
- Testing is conducted on a separate dataset to evaluate the model's accuracy, ensuring its ability to perform in real-world scenarios.

5. Disease Detection:

Once trained, the model can classify new images into predefined categories.

- If the leaf is healthy, the system confirms its state.
- If diseased, the system identifies the specific disease and provides confidence levels or accuracy percentages to enhance user trust in the diagnosis.
- An alert system may also recommend immediate actions or preventive measures based on the diagnosis.

6. User Interface:

The final results are presented on a **user-friendly frontend interface** designed for simplicity and accessibility.

- **Actionable Insights:** Users receive clear and concise recommendations for treatment or prevention.
- **Graphical Representation:** Results are visualized using charts or labeled images to help users understand the diagnosis.

3.2 Requirement Specification

To implement the proposed solution, the following tools and technologies are required:

3.2.1 Hardware Requirements:

- **Processor**: A minimum of **Intel Core i5** or higher is required for optimal performance. The processor should have enough power to handle complex computations, particularly during the training phase of machine learning models and processing large datasets. A higher processor specification will speed up training times and improve overall system responsiveness.
- **RAM**: At least **8 GB** or more of RAM is necessary for running multiple applications and processes concurrently. Adequate memory ensures that the system can handle the large volumes of data involved in training and testing the plant disease detection models, without running into memory bottlenecks. This also helps when performing image preprocessing and running inference tasks.
- **Storage**: **256 GB SSD** or higher is recommended for quick access to data and efficient model training. The SSD ensures fast data retrieval and storage, essential for handling high-resolution images and large datasets associated with plant disease detection. A higher storage capacity allows the system to store large image datasets, model checkpoints, and other relevant data without running out of space.
- **Graphics**: A **dedicated GPU** like the **NVIDIA GTX 1050** or higher is essential for deep learning tasks. The GPU accelerates the training of neural networks, reducing the time required to train models on large datasets. It is particularly important for image processing and feature extraction in deep learning models, ensuring fast and efficient training cycles.

- **Camera**: A **high-resolution camera** is necessary to capture detailed images of plant leaves. The camera should be capable of taking clear images to allow for accurate diagnosis of plant diseases. High-quality images are essential for feeding into the deep learning models, ensuring that the predictions made by the system are reliable and accurate.

3.2.2 Software Requirements:

- **Operating System**: The software environment can be set up on several operating systems, including **Windows 10/11**, **Linux (Ubuntu 20.04 or higher)**, or **macOS**. These operating systems provide the necessary stability and support for the required software tools and libraries. Linux, in particular, is preferred by many for its flexibility and performance in machine learning tasks, while macOS and Windows are also suitable options for development and testing.
- **Programming Languages**: **Python 3.x** is the primary language used for implementing machine learning models in this project. Python offers a rich ecosystem of libraries and frameworks that simplify tasks like image processing, model training, and evaluation. It is widely adopted in the machine learning community, providing extensive support for deep learning frameworks and data manipulation tools.
- **Libraries and Frameworks**:
    - **Deep Learning Framework**: To build and train the plant disease detection models, **PyTorch** is used in this study. PyTorch is a popular deep learning framework known for its flexibility, dynamic computation graph, and user-friendly API. Other frameworks like **TensorFlow** could be used as alternatives, but PyTorch is chosen here for its ease of use and robust support for custom model development.
    - **OpenCV**: OpenCV is used for **image preprocessing** tasks such as resizing, cropping, normalization, and augmentation. It is essential for preparing the plant images before feeding them into the deep learning models, ensuring that the data is in the correct format for accurate predictions.

  - **NumPy, Pandas, and Matplotlib**: These libraries are crucial for data manipulation, analysis, and visualization. **NumPy** provides support for efficient numerical operations on large arrays, while **Pandas** allows for easy manipulation of datasets. **Matplotlib** is used for visualizing data insights and model performance, and **Seaborn** enhances the data visualization capabilities for more detailed and aesthetically pleasing plots.

- **IDE/Editor**: **Jupyter Notebook** or **Visual Studio Code** are recommended integrated development environments (IDEs) for writing and running code. Jupyter Notebook provides an interactive environment that is particularly suited for machine learning experiments and prototyping. **Visual Studio Code** offers a powerful, lightweight editor with support for Python and machine learning libraries, making it a great choice for full-scale development.
- **Frontend Development**: The **Streamlit** framework is used for developing the frontend of the plant disease detection system. Streamlit allows for the rapid development of interactive web applications with minimal effort, making it ideal for building a user-friendly interface that integrates seamlessly with the machine learning model. This enables users, such as farmers, to upload images and view predictions in real-time.
- **Version Control**: **Git** and **GitHub** are used for managing the source code of the project. Git allows for efficient tracking of code changes, version control, and collaboration between team members. GitHub serves as a cloud-based repository to store the code, manage contributions, and facilitate collaboration. Using version control ensures that the codebase remains organized and can be easily updated or rolled back as needed during the development process.

CHAPTER 4

Implementation and Result

4.1 Snap Shots of Result:

1.Code initialization a Convolutional Neural Network (CNN) using Keras

```
cnn = tf.keras.models.Sequential()
```

```
cnn.compile(optimizer=tf.keras.optimizers.Adam(
    learning_rate=0.0001),loss='categorical_crossentropy',metrics=['accuracy'])
```

5]

```
cnn.summary()
```

6]

```
Model: "sequential"
```

```
cnn.summary()
```

[16]

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 128, 128, 32)	896
conv2d_1 (Conv2D)	(None, 126, 126, 32)	9,248
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0

conv2d_1 (Conv2D)	(None, 126, 126, 32)	9,248
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
conv2d_2 (Conv2D)	(None, 63, 63, 64)	18,496
conv2d_3 (Conv2D)	(None, 61, 61, 64)	36,928
max_pooling2d_1 (MaxPooling2D)	(None, 30, 30, 64)	0
conv2d_4 (Conv2D)	(None, 30, 30, 128)	73,856
conv2d_5 (Conv2D)	(None, 28, 28, 128)	147,584
max_pooling2d_2 (MaxPooling2D)	(None, 14, 14, 128)	0
conv2d_6 (Conv2D)	(None, 14, 14, 256)	295,168
conv2d_7 (Conv2D)	(None, 12, 12, 256)	590,080
max_pooling2d_3 (MaxPooling2D)	(None, 6, 6, 256)	0
conv2d_8 (Conv2D)	(None, 6, 6, 512)	1,180,160
conv2d_9 (Conv2D)	(None, 4, 4, 512)	2,359,808
max_pooling2d_4 (MaxPooling2D)	(None, 2, 2, 512)	0
dropout (Dropout)	(None, 2, 2, 512)	0
flatten (Flatten)	(None, 2048)	0
dense (Dense)	(None, 1500)	3,073,500
dropout_1 (Dropout)	(None, 1500)	0
dense_1 (Dense)	(None, 38)	57,038

Total params: 7,842,762 (29.92 MB)

Trainable params: 7,842,762 (29.92 MB)

Non-trainable params: 0 (0.00 B)

Fig2. Summary of Convolutional Neural Network (CNN) using Keras

This code defines a Convolutional Neural Network (CNN) for image classification into 38 categories. The model starts with convolutional layers that apply filters to extract features like edges and textures from input images. These are followed by pooling layers to reduce the spatial size, which helps in minimizing computational complexity. Dropout layers are added to prevent overfitting by randomly deactivating some neurons during training, ensuring better generalization. The final dense layer uses the softmax activation function to

output probabilities for each of the 38 classes. The model is compiled using the Adam optimizer for efficient training, with categorical cross-entropy as the loss function and accuracy as the performance metric. This structure ensures the model is robust, efficient, and capable of identifying patterns in image data accurately.

2.Visualization of Accuracy Result

```
epochs = [i for i in range(1,11)]
plt.plot(epochs,training_history.history['accuracy'],color='brown',label='Training Accuracy')
plt.plot(epochs,training_history.history['val_accuracy'],color='green',label='Validation Accuracy')
plt.xlabel('No. of Epochs')
plt.title('Visualization of Accuracy Result')
plt.legend()
plt.show()
```

[24]

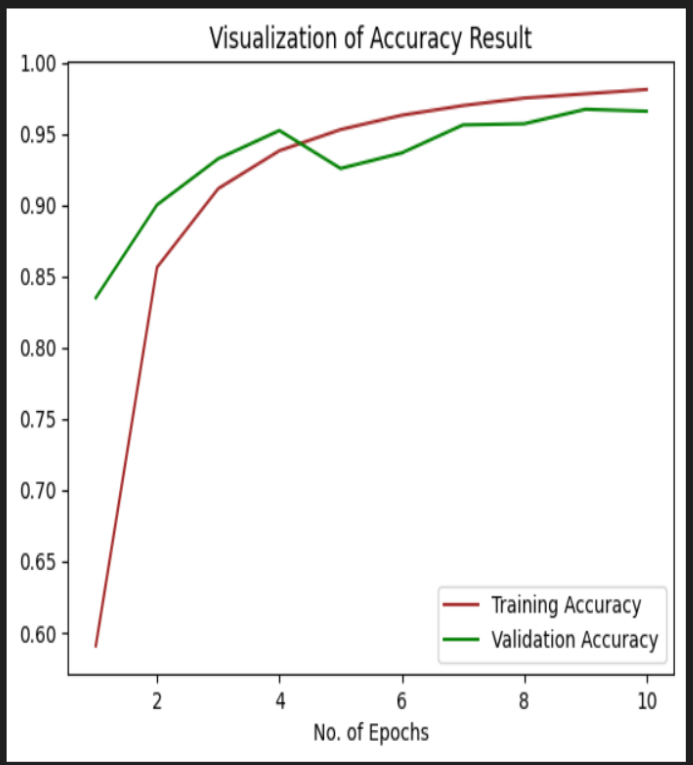

Fig3. Visualization of Accuracy Result

This graph shows two important aspects of the model's performance: Training Accuracy and Validation Accuracy over a span of 10 epochs (iterations).

- **Training Accuracy (Brown Line)**: This represents how well the model is performing on the data it has seen during training. As the model trains over multiple epochs, this line should ideally increase, showing the model is improving its predictions on the training data.
- **Validation Accuracy (Green Line)**: This line shows how well the model is performing on data it has never seen before, i.e., the validation data. This helps assess if the model is generalizing well or just memorizing the training data. A steady or increasing validation accuracy suggests good generalization.

3. Performance metrics for a classification model: the Confusion Matrix and the Classification Report.

```
from sklearn.metrics import confusion_matrix,classification_report
cm = confusion_matrix(Y_true,predicted_categories)
```

```
print(classification_report(Y_true,predicted_categories,target_names=class_name))
```

```
                                                precision    recall  f1-score   support

                             Apple___Apple_scab       0.96      0.97      0.97       504
                              Apple___Black_rot       0.98      0.98      0.98       497
                       Apple___Cedar_apple_rust       0.94      1.00      0.97       440
                              Apple___healthy         0.94      0.97      0.95       502
                          Blueberry___healthy         0.98      0.98      0.98       454
         Cherry_(including_sour)___Powdery_mildew       0.99      0.98      0.98       421
                Cherry_(including_sour)___healthy       0.98      0.98      0.98       456
Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot       0.96      0.90      0.93       410
                      Corn_(maize)___Common_rust_       0.99      0.99      0.99       477
              Corn_(maize)___Northern_Leaf_Blight       0.93      0.97      0.95       477
                       Corn_(maize)___healthy           0.99      1.00      0.99       465
                          Grape___Black_rot            1.00      0.94      0.97       472
                    Grape___Esca_(Black_Measles)       0.95      1.00      0.97       480
      Grape___Leaf_blight_(Isariopsis_Leaf_Spot)       0.99      1.00      0.99       430
                            Grape___healthy            0.99      1.00      1.00       423
         Orange___Haunglongbing_(Citrus_greening)       0.97      1.00      0.98       503
                         Peach___Bacterial_spot        0.97      0.95      0.96       459
                              Peach___healthy          0.97      0.99      0.98       432
                  Pepper,_bell___Bacterial_spot        0.96      0.96      0.96       478
                         Pepper,_bell___healthy        1.00      0.88      0.94       497
                          Potato___Early_blight        0.98      0.97      0.98       485
                           Potato___Late_blight        0.89      0.99      0.94       485
                              Potato___healthy         0.96      0.97      0.96       456
...
                                            accuracy                           0.97     17572
                                           macro avg       0.97      0.97      0.97     17572
                                        weighted avg       0.97      0.97      0.97     17572
```

Fig4. Performance metrics for a classification model: the Confusion Matrix and the Classification Report.

- The model achieved an impressive **overall accuracy of 97%**, demonstrating strong performance across a variety of plant disease classifications. Most of the classes show high precision, recall, and F1-scores, indicating that the model is reliably distinguishing between different plant diseases. For instance, **Grape___healthy** and **Corn___healthy** achieved perfect recall (1.00), meaning the model correctly identified all healthy instances in these classes, while maintaining high precision as well.
- Although some classes, such as **Potato___Late_blight**, have slightly lower precision (0.89), the model maintains a very high recall (0.99), minimizing the number of false negatives. This ensures that the model is highly effective at detecting the presence of diseases even when it makes occasional misclassifications.
- The **macro average** and **weighted average** F1-scores are both **0.97**, indicating balanced performance across all classes, with larger classes having a slightly more significant influence on the overall result.
- Overall, the model shows excellent classification capabilities with strong generalization across a diverse set of plant diseases, making it suitable for practical use in plant health monitoring applications.

4. Heatmap Visualizing the confusion matrix

```
    plt.figure(figsize=(40, 40))
    sns.heatmap(cm,annot=True,annot_kws={"size": 10}, cmap='tab10')

    plt.xlabel('Predicted Class',fontsize = 30)
    plt.ylabel('Actual Class',fontsize = 40)
    plt.title('Plant Disease Prediction Confusion Matrix',fontsize = 25)
    plt.show()
```

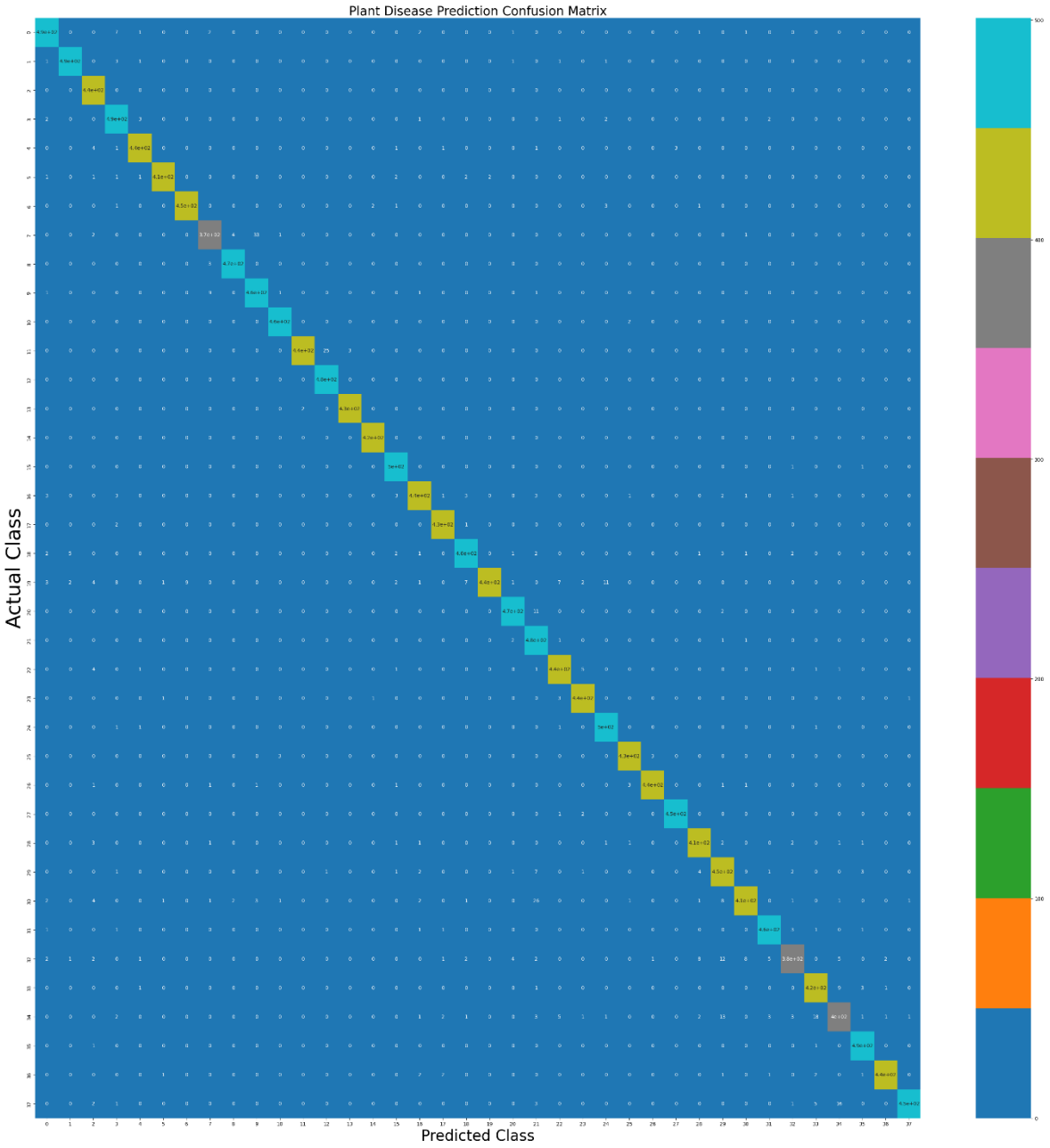

Fig5.Heatmap Visualizing the confusion matrix

The heatmap visualizes the confusion matrix, which shows how well the model's predictions match the actual classes. The rows represent the true labels (actual classes), and the columns represent the predicted labels. The numbers inside each cell indicate how many predictions fall into each category. The colors help differentiate between correct and incorrect predictions, with brighter colors typically indicating higher values. This heatmap allows us to easily see where the model performs well (along the diagonal) and where it makes errors (off-diagonal).

5. User Interface of Plant Disease Detection System

Fig 6.1. Home Page of Plant Disease Detection System

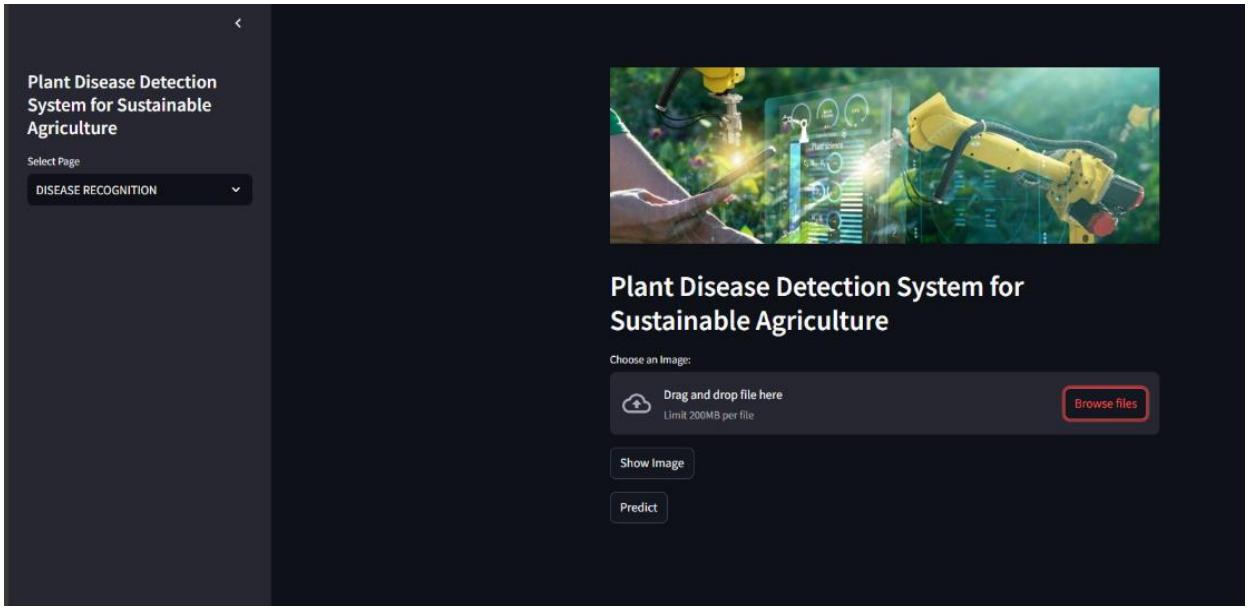

Fig 6.2. Disease Recognition Page of Plant Disease Detection System

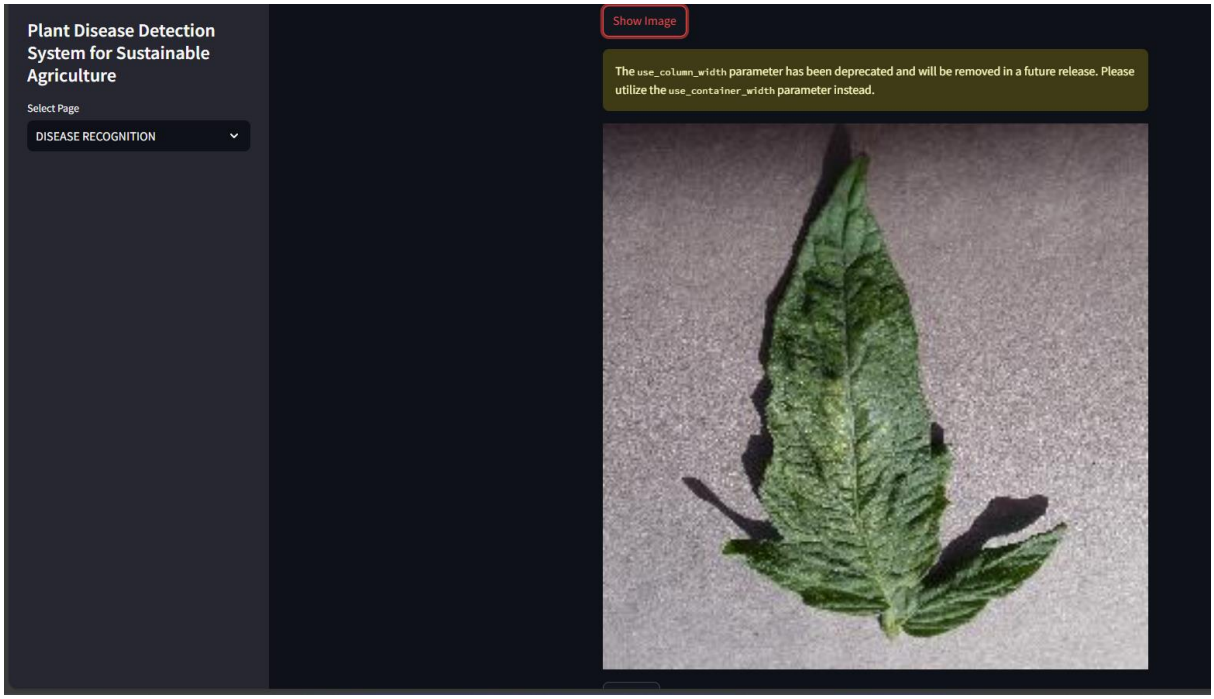

Fig 6.3. Show Image Feature of Disease Recognition Page of Plant Disease Detection System

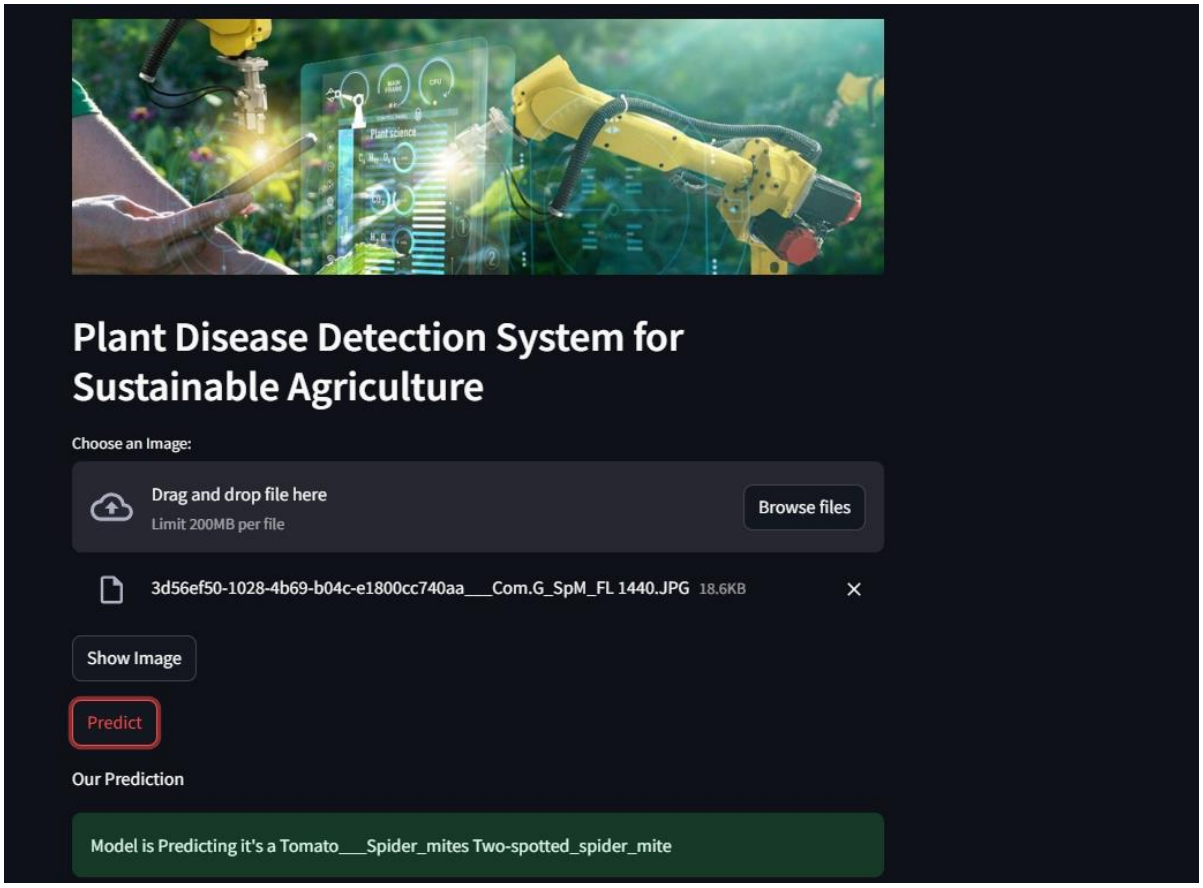

Fig 6.4. Predict Feature of Disease Recognition Page of Plant Disease Detection System

- The User Interface of plant disease detection system shows two pages namely a home page and a disease recognition page.
- The home page is just show information of system. The disease recognition page contains two features: Show Image and Predict.
- When we upload an image we can check whether proper image is uploaded or not by using show image feature.
- And then predict feature predicts disease present on plant.

4.2GitHub Link for Code:

GitHub Link for project is given as: **sadneya145/plant_disease_detection**

CHAPTER 5

Discussion and Conclusion

5.1 Future Work:

1. **Enhancing the Dataset:**
    - Expand the dataset to include a wider variety of plants, diseases, and environmental conditions.
    - Include multi-modal data inputs such as weather conditions, soil quality, and pest activity to improve the model's robustness and accuracy.
    - Increase diversity in the dataset to handle variations like lighting, angles, and overlapping symptoms effectively.
2. **Improving Model Accuracy:**
    - Test and implement advanced deep learning architectures such as EfficientNet or Transformers for better performance.
    - Use data augmentation techniques to simulate real-world conditions and reduce overfitting.
    - Experiment with ensemble learning methods to combine predictions from multiple models for increased reliability.
3. **Integration of Frontend and Backend:**
    - Develop a user-friendly frontend interface for better user interaction.
    - Integrate the backend with the frontend for real-time disease detection, analysis, and monitoring.
    - Implement authentication and security protocols to protect user data and ensure safe access to the application.
4. **Deployment and Real-World Testing:**
    - Deploy the model on cloud platforms or mobile devices for real-time field usage.

- Test the system in real-world agricultural environments to validate its performance and gather user feedback.
- Optimize the application for low-resource devices, ensuring accessibility for farmers in rural areas.

5. **Recommendation System Development:**
   - Add a recommendation feature to suggest treatment options and preventive measures for detected diseases.
   - Provide insights and tips based on historical data and real-time monitoring to help farmers manage their crops effectively.
6. **Language and Regional Support:**
   - Implement multilingual support for better accessibility across different regions.
   - Customize the system for specific regional crops and diseases to ensure broader adoption and relevance.
7. **Continuous Learning and Updates:**
   - Enable the model to update itself with new data, improving accuracy over time.
   - Incorporate feedback from end-users to address unresolved issues and refine the system further.
8. **User-Friendly Mobile Application**

   A mobile application could be developed to make the system more accessible to farmers.
   - Features such as multilingual support, regional customization, and voice-based interaction would cater to a broader audience, including those with limited technological literacy.
9. **Integration with Government and Agribusiness Initiatives**
   - The system can be linked with government schemes and agribusinesses to provide subsidies, guidance, and market access to farmers. This collaboration could make the solution more sustainable and widely adopted.
10. **Multimodal Data Inputs**
    - Introducing multimodal data inputs, such as images, text, and environmental sensor data, would make the system more robust and capable of handling complex scenarios.
    - For instance, combining leaf image analysis with textual descriptions of symptoms could improve disease diagnosis.

5.2 Conclusion:

As there are many fertilizers are present nowadays which are good for preventing plant disease but due to less knowledge the farmers use it in a lot of amount for saving crops which lead to deteriorating human health. Also farmers pay a huge amount to employ a person to manually check plants and crops which are healthy and diseased. So this a model which is better than human in terms of time and cost. The model has capability to reduce the burden of many agricultural practitioner who are well trained and educated.

Thus,this project has made a significant contribution to agriculture by developing a system to detect plant diseases using machine learning. It helps farmers identify crop issues early, reducing losses and improving productivity. The use of image processing and advanced models like CNN has shown promising results in accurately identifying diseases.

The project highlights how technology can transform traditional farming practices, making them more efficient and sustainable. While there is still room for improvement, such as expanding datasets and integrating user-friendly interfaces, the current system provides a solid foundation for future enhancements.

In summary, this project bridges the gap between technology and agriculture, offering a practical solution for healthier crops and better farming outcomes.

REFERENCES

[1]. Gaurav Verma, Charu Taluja, Abhishek Kumar Saxena “Vision Based Detection and Classification of Disease on Rice Crops Using Convolutional Neural Network” ,2019

[2]. Nikhil Shah1, Sarika Jain2 “Detection of Disease in Cotton Leaf using Artificial Neural Network”,2019

[3]. Ch. Usha Kumari “Leaf Disease Detection: Feature Extraction with K-means clustering and Classification with ANN”,2019

[4]. S. D.M., Akhilesh, S. A. Kumar, R. M.G. and P. C., "Image based Plant Disease Detection in Pomegranate Plant for Bacterial Blight," 2019 International Conference on Communication and Signal Processing (ICCSP), 2019, pp. 0645-0649, doi: 10.1109/ICCSP.2019.8698007.

[5]. Kumar, M., Gupta, P., Madhav, P., & Sachin, “Disease Detection in Coffee Plants Using Convolutional Neural Network”,2020