**PlantGuard:** **AI Disease Detection & Real-Time Plant Advice**

**Team: Sridhar Kandi, Abishek Challa**

**Abstract:**

Gardeners, plant enthusiasts, and farmers often face challenges in identifying diseases in their plants early, which can lead to reduced yields and increased reliance on chemicals. Our project aims to develop a smart system that identifies plant diseases and provides real-time assistance to users. The system will utilize Convolutional Neural Networks (CNN) to accurately detect diseases from images of plant leaves. To offer personalized advice and support, we plan to integrate a LangChain-based LLaMa Large Language Model (LLM), fine-tuned to meet the specific needs of gardeners, home growers, and farmers alike. This will enable the system to provide instant, tailored recommendations, empowering users with actionable insights for managing plant health effectively. Users will be able to upload images through a user-friendly interface, and the system will deliver immediate results. Additionally, a chatbot powered by the fine-tuned LangChain LLaMa LLM will be available to answer queries about plant diseases and general care tips in simple language. The system will store all disease detection results and chatbot interactions in a MongoDB database for easy retrieval, allowing users to track the health of their plants over time.

**Expected Outcomes:**

* Farmers will be able to upload plant leaf images through a user-friendly interface and receive instant feedback on potential diseases.
* A user-friendly LLM chatbot will assist farmers by providing advice on better crop healthcare, real-time updates, and information on relevant government schemes.
* The system will store all data in a database, allowing easy access to past results and conversations.
* The web interface will enable easy access to the disease detection system and chatbot, making it convenient for farmers to interact with the platform.

**Objectives and Relevance:**

The primary objective of this project is to develop an innovative, AI-powered system which can assist users in detecting plant diseases early and provide personalized, real-time support. The system will use cutting-edge technologies like Convolutional Neural Networks (CNNs) to accurately identify diseases from images of plant leaves, paired with a LangChain-based LLaMa Large Language Model (LLM) fine-tuned for the agricultural domain. By combining these technologies, the system aims to empower users with immediate, actionable insights, which will improve crop health management and increase plant productivity.

**Relevance for the industry:**

* The system helps farmers and home gardeners detect plant diseases early, improving plant health and yields while reducing the need for excess chemicals.
* Users can easily upload plant images and get real-time results, with a chatbot offering tailored advice and support on plant care, pest control, and more.
* The system is affordable and accessible, making expert-level disease detection and personalized recommendations available to everyone.
* It adapts to different plants and regions, benefiting both home growers and large-scale farmers.
* By supporting sustainable and precision growing, the system promotes better decisions, less waste, and improved productivity for all users.

**References to show from where the above information has come**

* Revolutionizing Farming: GAN-Enhanced Imaging, CNN Disease Detection, and LLM Farmer Assistant

<https://ieeexplore.ieee.org/document/10486501>

* PlantDoc: A Dataset for Visual Plant Disease Detection

<https://arxiv.org/abs/1911.10317>

* AI for crop production – Where can large language models (LLMs) provide substantial value?

<https://www.sciencedirect.com/science/article/pii/S0168169924003156>

**Data sources we have explored steps we have followed to get full access to the data**:

For our project, we employed a range of tools and approaches to identify and use the PlantDoc dataset. To learn about the dataset, we began by reading academic articles and browsing reliable websites. We also explored Google Datasets to determine what data sources we could use for our project. To broaden our search, we checked out the Awesome-Public-Datasets project on GitHub, which contains a significant number of publicly available datasets. Finally, we obtained the PlantDoc dataset from a GitHub site. This repository contained detailed information on the dataset's structure and how to use it. The PlantDoc dataset is licensed under Creative Commons Attribution 4.0 International (CC BY 4.0), which allows us to use it without restriction as long as we credit the original authors. This way, we made sure we followed all legal and ethical guidelines for using the dataset.

**Things that cannot be done using few prompts:**

In order to reliably identify plant diseases, the research entails advanced image analysis employing Convolutional Neural Networks (CNNs), which calls for processing beyond basic text prompts. Furthermore, it incorporates an improved LangChain-based LLaMa model to give farmers tailored, specific to the situation assistance, requiring more than simple prompt-based interactions. The solution also incorporates sophisticated data management using a MongoDB database for storing and retrieving disease detection.

results, as well as real-time application integration. These features go beyond what prompts alone can accomplish.

**Dataset Exploration and Image Classification for Plant Diseases:**

A screenshot of a computer

Description automatically generated

Our primary focus was on getting to know about the data/ images available for various plant diseases in our dataset, which is crucial for training our Convolutional Neural Networks (CNN).

We successfully implemented a Python function to count the number of images in each class within our dataset. We found detailed count of images for various plant diseases as in the screenshot.

The dataset consists of various plant leaf images categorized into different classes, each representing a specific disease or plant type. The CNN model defined for classification includes several convolutional layers with ReLU activation and max pooling, followed by fully connected layers. The final layer uses a SoftMax activation function for multi-class classification, corresponding to the number of classes in the dataset.

A screenshot of a computer program

Description automatically generated

We have defined a Convolutional Neural Network (CNN) model for classification using TensorFlow and Keras. The model consists of several convolutional layers with ReLU activation and max pooling, followed by fully connected layers. Currently, training this model for one epoch takes approximately 15 minutes. To improve the training speed, we are considering upgrading to Google Colab Pro, which offers enhanced computational resources. This upgrade is expected to significantly reduce training time and improve efficiency in our model development process. We have not yet run the model but plan to do so after acquiring the necessary resources.

In the coming weeks, we plan to evaluate our plant dataset using multiple pretrained models, including ResNet50, MobileNet, and AlexNet. By testing these models, we aim to compare their performance and identify which model offers the highest accuracy for detecting plant diseases. In addition to CNN models, we will also explore integrating Large Language Models (LLMs) to see if they can further improve our system's performance, particularly in providing personalized advice.

To enhance the model’s accuracy, we plan to augment our dataset by applying various data augmentation techniques, such as rotation, flipping, and scaling. This will effectively increase the size and diversity of our training data, which is expected to improve the robustness and generalization capabilities of our models. By conducting these tests, we hope to optimize the model's performance and ensure it is well-suited for real-world use.

**Exploring LLM Integration with the model:**

A screenshot of a chat

Description automatically generated

As we are new to the concept of **Retrieval-Augmented Generation (RAG)**, we wanted to explore and experiment with it using a small dataset. We started by using ChromaDB as a vector database to store and retrieve embeddings, and LlamaIndex to structure and index large datasets. This allowed for efficient retrieval of relevant documents. We integrated these tools with the OpenAI API for generating contextually relevant responses based on the retrieved information.

We have successfully completed the setup of the OpenAI API account and generated an API key named `OPENAI\_LLM` for our project. As the next steps, we will ensure the API key is securely stored within our project’s environment to maintain proper security and prevent unauthorized access. Once securely stored, we will use the key to make the necessary API calls to OpenAI services, enabling the language model functionalities within our project.

A screenshot of a chat

Description automatically generated

We have realized that to successfully fetch data through API calls, we need to maintain sufficient credits in our account. Currently, our credit balance stands at $4.92. It's important to note that once the balance reaches $0, the API requests will stop working.



A screenshot of a computer code

Description automatically generated

We have made progress on the Large Language Model (LLM) integration. To begin with, we experimented with the GPT-2 medium model, which offers general solutions for plant diseases. While it provides a basic framework for disease-related queries, we are exploring more specific, fine-tuned LLMs that are tailored to plant disease identification. Our goal is to improve the system's ability to provide personalized and contextually relevant advice to users.

A screenshot of a computer

Description automatically generated

Additionally, we are considering using the Retrieval-Augmented Generation (RAG) technique. This approach will allow us to combine a base model with a retrieval mechanism to fetch relevant plant disease information from external sources, further improving the quality of advice provided by the system. We are still in the process of deciding whether to fine-tune an existing model for this task or use RAG with an off-the-shelf model.

**Introduction:**

***Summarizations and Understandings form the Reference Papers***

**Reference 1:**

**Plant Disease Detection with Deep Learning and Feature Extraction Using Plant Village**

This paper talks about using deep learning to detect diseases in plants, which is very important because crop diseases can greatly affect the food supply. Farmers, especially in less developed areas, often don’t have access to high-tech tools that help them identify these diseases early. With advancements in technology like smartphones and artificial intelligence (AI), new tools are being developed to help farmers identify plant diseases more easily. The paper focuses on Convolutional Neural Networks (CNNs), a popular type of deep learning model that works well with images.

In this, the researchers used the PlantVillage dataset, which contains many images of different plant diseases. They tested several CNN models like ResNet 50, Google Net, and VGG-16 to see which one performed the best at detecting diseases. These models use a method called deep feature extraction, which helps the computer learn important patterns from the images. After extracting the features, they used two different classifiers: SVM (Support Vector Machine) and k-Nearest Neighbors (KNN) to identify the diseases.

The results showed that SVM was the most accurate at classifying the diseases. The study also found that feature extraction worked better than transfer learning for this task. Transfer learning is when you take a pre-trained model and fine-tune it for a new task. While transfer learning is useful, in this case, deep feature extraction with SVM gave better results in terms of both speed and accuracy.

For the future, the researchers suggest collecting data from more varied environments. The PlantVillage dataset is good, but it contains images with simple backgrounds, which doesn’t represent real-world farming conditions. By studying plants in different and more challenging environments, they hope to improve the model’s ability to detect diseases more accurately, helping farmers prevent crop losses more effectively.

**Reference 2:**

**An Analysis of Plant Diseases Identification Based on Deep Learning Methods**

This paper looks at two different models, Faster R-CNN and YOLOv3, which are used for detecting multiple diseases in apple leaves. These models are part of a type of deep learning called object detection, which helps not only identify what the disease is but also where it is located on the leaf. The study aims to make disease detection faster and more accurate because plant diseases can spread quickly, and farmers need timely interventions.

Faster R-CNN and YOLOv3 are popular object detection models. The Faster R-CNN model works in two stages: first, it generates several regions where a disease might be present, and then it classifies each region. On the other hand, YOLOv3 does everything in one step, which makes it faster. The researchers used a dataset with images of five different apple leaf diseases taken in real-world environments, not just in controlled lab settings.

The study found that Faster R-CNN was better at identifying small diseases that were in the early stages, but it was slower, with a speed of only 13.9 frames per second (FPS). On the other hand, YOLOv3 was much faster, detecting diseases at 69.0 FPS, though it was slightly less accurate than Faster R-CNN. The researchers concluded that each model has its strengths: Faster R-CNN is more accurate but slower, while YOLOv3 is faster but less precise. For real-world applications where speed is important, like in large farms, YOLOv3 might be more useful, but Faster R-CNN would be better for early-stage disease detection.

The paper suggests that both models could be further improved. For example, Faster R-CNN could be optimized to run faster, and YOLOv3 could be tweaked to improve its accuracy. Future research should also focus on adapting these models for portable devices like smartphones, which would make it easier for farmers to detect plant diseases in the field.

Key Points:

* Faster R-CNN is more accurate, especially for small, early-stage diseases.
* YOLOv3 is much faster and could be more useful in real-time applications.
* Both models could be improved, and future research should focus on making them usable on portable devices for real-world use.

**Reference 3:**

**Large Language Models and Foundation Models in Smart Agriculture: Basics, Opportunities, and Challenges**

This paper explores how Foundation Models (FMs), which are large, pre-trained AI models, can be applied in agriculture. Traditional Machine Learning (ML) and Deep Learning (DL) models have made big advances in farming, helping with tasks like crop management and livestock monitoring. However, these models need large datasets and are usually designed for one specific task. Foundation Models are different because they are trained on huge amounts of data across many fields and can be fine-tuned with very little task-specific data. This makes them more flexible and efficient.

The paper reviews different types of foundation models, like Language Models (FMs), Vision Models (FMs), and Multimodal Models (FMs), and how they could be used in agriculture. For example, Language FMs could be used to analyze text data, such as weather reports or research articles, to provide better farming recommendations. Vision FMs could help with tasks like detecting diseases in plants or monitoring the health of livestock. Multimodal FMs can work with both text and images, making them very versatile for agricultural applications.

The paper also talks about the challenges of using these large models in agriculture. One issue is that agriculture data can be very different depending on the region or climate, so these models need to be carefully trained and tested to work well in different conditions. Another challenge is deploying these models in real-world environments, where they need to work reliably even with limited internet access or computing power.

For the future, the paper suggests building large agricultural datasets that cover many different types of crops, animals, and farming conditions. This would help in training Foundation Models specifically for agriculture, making them even more effective and useful for farmers around the world.

**Reference 4:**

**Plant Disease Detection and Classification by Deep Learning**

This paper reviews how deep learning (DL) has been used to detect and classify plant diseases. Deep learning is part of machine learning and has become very popular because it can automatically learn features from data, like images of diseased plants, without needing humans to manually label them. The paper discusses various DL models that have been used for this task, including Convolutional Neural Networks (CNNs), which are especially good at working with images.

One of the main challenges with current DL models is that they mostly use simple datasets like the PlantVillage dataset, which features plant images with plain backgrounds. While this helps the models perform well in tests, it doesn’t reflect the complexity of real-world environments, where plants grow in messy fields with varying lighting and other conditions. The paper suggests that future models need to work in these more complex environments if they are going to be useful for real-world farming.

The paper also talks about using new technologies like hyperspectral imaging and multispectral imaging to detect plant diseases even before symptoms are visible. This could allow farmers to take action earlier and prevent the spread of disease. The authors recommend designing models that can track the entire lifecycle of a plant disease, from the early stages to later, more severe stages.

In the future, the paper suggests expanding datasets to include more images from real-world farming environments. It also suggests improving DL models to handle different conditions like lighting, disease severity, and environmental changes. By doing this, these models can become more practical for farmers who need to detect and manage plant diseases effectively.

**Methods:**

we utilized the PlantVillage dataset available through TensorFlow Datasets (TFDS). This dataset is a large collection of labeled images for training machine learning models to detect plant diseases. It contains over 54,000 images of healthy and diseased crop leaves, organized into 38 distinct classes. The dataset is freely available for research and educational purposes through TensorFlow Datasets.

**Dataset Link:** <https://www.tensorflow.org/datasets/catalog/plant_village>

There are no explicit legal limitations regarding the use of this dataset, as it is open access for non-commercial use. However, the dataset contains no personally identifiable health information (PIHI), so it is not subject to healthcare privacy laws like HIPAA (Health Insurance Portability and Accountability Act). Since this is agricultural data, there are no restrictions like those found with real human health data, where regulations such as HIPAA would apply to protect sensitive information.

**Convolutional Neural Network (CNN) Approach**

We implemented a CNN to classify images of plant leaves into 38 categories, including healthy and diseased states. The model consists of three convolutional layers (with 32, 64, and 64 filters) followed by max-pooling layers to reduce dimensionality and extract key features. A fully connected layer with 64 units and a softmax output layer predicts the class probabilities.

We applied data augmentation (flips, rotations, shifts) to enhance generalization, used the Adam optimizer for training, and employed sparse categorical cross-entropy as the loss function. The dataset was split into training, validation, and test sets (80/10/10), with early stopping and model checkpointing to avoid overfitting.

After training for a few epochs, the model achieved a test accuracy of **86.26%**. Performance evaluation using accuracy, confusion matrix, and classification report showed strong results in detecting plant diseases.

A screenshot of a computer code

Description automatically generated

**Hugging Face ViT Model Approach:**

we will be using the Vision Transformer (ViT) model, specifically the fine-tuned version of the google/vit-base-patch16-224 from Hugging Face. This model has been pre-trained for image classification tasks, but we plan to further fine-tune it on the PlantVillage dataset to improve its accuracy for plant disease detection.

The ViT model processes images by splitting them into patches, then applies a transformer to classify the image based on its visual features. Using this approach, the model has already achieved an impressive validation accuracy of \*\*99.33%\*\* during prior fine-tuning. However, for our specific application, we will retrain the model using the PlantVillage dataset to optimize its performance for distinguishing between the 38 plant disease classes.

After training, we will evaluate the model on a separate test set to assess its accuracy and generate a classification report along with a confusion matrix to provide detailed insights into the model's predictions.This fine-tuning process will allow us to create a highly effective image classifier for plant disease detection using the latest transformer-based techniques.

We will also train the ResNet50 model on the PlantVillage dataset for plant disease classification, using a batch size of 16 and the Adam optimizer. The model will be evaluated using accuracy metrics on both the validation and test sets. Additionally, we will train the ViT model under similar conditions. At the end of the training, we will compare the performance of both models using classification reports, confusion matrices, and test accuracy to determine the best model for plant disease detection. The results will guide us in selecting the most effective approach for this task.

A computer code with text

Description automatically generated

Additionally, I have tested the OpenAI GPT model via API and observed that it provides generic answers. We plan to fine-tune it with the plant disease dataset for more accurate, domain-specific responses. Moreover, we need to explore more in the LLM part, particularly how it answers questions using similarity search algorithms like FAISS to improve response relevance.

Method related links:

<https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2023.1308528/full>

<https://publications.eai.eu/index.php/IoT/article/view/4578>

<https://escholarship.org/uc/item/18q4s6kh>

<https://huggingface.co/blog/fine-tune-vit>

<https://pmc.ncbi.nlm.nih.gov/articles/PMC10708852/>

### **References:**

1. An Analysis of Plant Diseases Identification Based on Deep Learning Methods

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10412967/>

1. AI in Agriculture: Transformative Use Cases, Success Stories, and Challenges <https://www.kubernet.dev/ai-in-agriculture-transformative-use-cases-success-stories-and-challenges/>

1. Large Language Models and Foundation Models in Smart Agriculture: Basics, Opportunities, and Challenges

<https://arxiv.org/html/2308.06668v4>

1. Plant Disease Detection and Classification by Deep Learning

<https://www.mdpi.com/2223-7747/8/11/468>

1. EfficientRMT-Net—An Efficient ResNet-50 and Vision Transformers Approach for Classifying Potato Plant Leaf Diseases

<https://pmc.ncbi.nlm.nih.gov/articles/PMC10708852/>

1. Uncovering bias in the PlantVillage dataset

<https://arxiv.org/abs/2206.04374>

1. Large Language Models for Crop Yield Prediction DOI: <https://doi.org/10.21203/rs.3.rs-4750823/v1>
2. Large Language Models and Foundation Models in Smart Agriculture: Basics, Opportunities, and Challenges <https://arxiv.org/html/2308.06668v4>
3. PlantPAD: a platform for large-scale image phenomics analysis of disease in plant science <https://pmc.ncbi.nlm.nih.gov/articles/PMC10767946/>
4. DiaMOS Plant: A Dataset for Diagnosis and Monitoring Plant Disease https://www.mdpi.com/2073-4395/11/11/2107

**GitHub Page:**

<https://github.com/sridhar051/Capstone_Data_606>

**Need to complete:**

1. Evaluate and compare different pretrained models (ResNet50, Huggingface Vit modle) using the PlantVillage dataset.
2. Explore fine-tuning the OpenAI GPT or other AI model for plant disease identification or finalize the use of similarity search algorithms for more accurate and specific recommendations.
3. Enhance system accuracy through hyperparameter tuning and model optimization.
4. Test and integrate the LLM-powered chatbot with plant disease classification for real-time, personalized advice.
5. Create an easy-to-use UI to use this.