**Plant Disease Classification**

**Statistical Machine Learning CSET211**

Submitted by:

(E23CSEU0992) Lakshit Gupta

(E23CSEU0995) Shruti Suman

(E23CSEU1009) Athira Ravi Pillai

Submitted to

**DR. NITIN ARVIND SHELKE**

**July-Dec 2024**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

A close-up of a logo

Description automatically generated

**INDEX**

|  |  |  |
| --- | --- | --- |
| Sr.No | Content | Page No |
| 1 | [Abstract](#Abstract_index) | 4 |
| 2 | [Introduction](#Introduction) | 4 |
| 3 | [Related Work](#related-work) | 4 |
| 4 | [Methodology](#methodology) | 5 |
| 5 | [Data Visualization and EDA](#data-visualization-and-eda) | 5 |
| 6 | [Hardware/Software Requirements](#hardware-software-requirements) | 5 |
| 7 | [Experimental Results](#experimental-results) | 6 |
| 8 | [Deployment](#deployment) | 6 |
| 9 | [GitHub Repository](#github-repository) | 6 |
| 10 | [Conclusion](#conclusion) | 6 |
| 11 | [Future Scope](#future-scope) | 6 |

1. **Abstract**
2. **Introduction**
3. **Related Work (If Any)**
4. **Methodology**
5. **Hardware/Software Required**
6. **Experimental Results**
7. **Conclusions**
8. **Future Scope**
9. **GitHub Link of Your Complete Project**

<https://github.com/Lakshit-Gupta/plant-disease-prediction-model>

**Plant Disease Classification Using Custom and Pretrained Deep Learning Models**

**[Abstract](#Abstract)**

Plant diseases significantly impact agricultural productivity, necessitating timely and accurate identification. This project focuses on classifying plant diseases using three deep learning models: two custom convolutional neural networks (CNNs) and an implementation of ResNet50, a well-known pretrained architecture. The dataset consists of labeled images of healthy and diseased plant leaves sourced from Kaggle. The models are evaluated on their performance, with a comparison of their accuracies and suitability for deployment. The final solution is optimized to aid farmers and agricultural experts in real-time disease detection.

**Introduction**

The agricultural sector is prone to substantial losses caused by plant diseases. Traditional methods of detecting these diseases are labor-intensive and error-prone. Leveraging machine learning and deep learning, this project aims to automate plant disease classification, providing a scalable and efficient solution.

**Key Objectives**:

1. Develop and compare multiple models for plant disease classification.
2. Optimize the performance of these models for real-world deployment.
3. Demonstrate the practicality of the solution via a GitHub-hosted repository and a project demo.

**Source**: Dataset obtained from Kaggle.

**Related Work**

Existing research employs both custom CNNs and pretrained models such as VGG16, ResNet, EfficientNet, and Swin Transformer for plant disease classification. While pretrained models often outperform custom ones, they require substantial computational resources. This project explores both approaches, balancing simplicity and performance.

**Methodology**

1. **Dataset Preparation**  
   The dataset comprises thousands of images of plant leaves labeled into categories such as healthy and diseased. Preprocessing steps include resizing, normalization, and data augmentation (rotation, flipping, and zooming) to enhance generalization.
2. **Models**
   * **Model 1**: A primary custom CNN with four convolutional blocks, each comprising Conv2D, batch normalization, max pooling, and dropout layers with L2 regularization.
   * **Model 2**: A secondary custom CNN, simpler than the first model, optimized for faster inference with L2 regularization.
   * **Model 3**: An implementation of ResNet50, pretrained on ImageNet, fine-tuned on the plant disease dataset.
3. **Training and Testing**
   * **Optimization**: Adam optimizer with categorical crossentropy as the loss function.
   * **Metrics**: Accuracy, precision, recall, and F1-score.
   * **Early Stopping and Learning Rate Reduction**: To prevent overfitting and improve convergence.
   * **Class Weights**: Computed using scikit-learn's compute\_class\_weight to handle imbalances.
4. **Evaluation**  
   The models were evaluated on a validation dataset using confusion matrices and classification reports. Model 1 achieved the highest accuracy, followed by ResNet50 and Model 2.

**Data Visualization and Exploratory Data Analysis (EDA)**

Data distribution was visualized using Python libraries like Matplotlib and Seaborn. EDA revealed significant class imbalances, which were addressed using class weighting during model training.

**Hardware/Software Requirements**

**Minimum Requirements**

* Processor: Intel/AMD Quad-core x64 processor.
* GPU: NVIDIA GPU, VRAM > 8 GB.
* RAM: Minimum 16 GB.

**Software Required**

* VS Code / Anaconda
* Jupyter Notebooks / PyCharm

**Experimental Results**

* **Model 1 (Primary Custom CNN)**: Validation accuracy of 94.78%, but with higher computational demands.
* **Model 2 (Secondary Custom CNN)**: Validation accuracy of 92.19%, demonstrating robustness with moderate training time.
* **Model 3 (ResNet50)**: Validation accuracy of 93.43%, optimized for faster deployment.

**Deployment**

The project was deployed as a web application using Streamlit. The user can upload images of plant leaves, and the deployed model predicts the disease category.

**GitHub Repository**

The complete project is hosted on GitHub, containing:

* Code files for all three models.
* Dataset used for training and testing.
* Project report and presentation.
* Deployment files.

**GitHub Link**: <https://github.com/Lakshit-Gupta/plant-disease-prediction-model>

**Conclusion**

This project demonstrates the effective use of deep learning for plant disease classification. Among the three models, the Primary Custom CNN outperformed others but at the cost of higher computational requirements. The Secondary Custom CNN offers a balance of performance and efficiency, making it suitable for real-time applications.

**Future Scope**

1. Enhance the dataset by including images from different environments.
2. Experiment with additional pretrained architectures like EfficientNet, Swin Transformer, and VGG16.
3. Develop a mobile application for wider accessibility.

This report is structured to address all milestones outlined in the marking scheme, emphasizing the systematic development, evaluation, and deployment of the project.