DATA SCIENCE PROJECT ON RICE LEAF DISEASE DETECTION

PROJECT ID: PRCP-1001-RICELEAF

PROJECT NAME: RICE LEAF DISEASE DETECTION

PROJECT TEAM ID: PTID-CDS-OCT-25-3251

CONTENTS

1. INTRODUCTION

1.1. PROJECT OVERVIEW

1.2. PROBLEM STATEMENT

1.3. OBJECTIVES

2. DATA ANALYSIS REPORT

2.1. DATASET DESCRIPTION

2.2. DATA PREPARATION AND SPLITTING

2.3. EXPLORATORY DATA ANALYSIS (EDA)

3. METHODOLOGY

3.1. DATA PREPROCESSING

3.2. DATA AUGMENTATION

3.3. MODEL ARCHITECTURE

4. RESULTS AND EVALUATION

4.1. MODEL TRAINING

4.2. MODEL PERFORMANCE

5. MODEL COMPARISON REPORT

6. REPORT ON CHALLENGES FACED

7. CONCLUSION

1. INTRODUCTION

1.1. PROJECT OVERVIEW

This report details the process and results for Project PRCP- 1001: Rice Leaf disease detection. The project involves analyzing a dataset of rice leaf images and developing a classification model to identify specific diseases.

1.2. PROBLEM STATEMENT

The objective is to create a model capable of classifying images of rice plant leaves into three major disease categories: Bacterial leaf blight, Brown spot, and Leaf smut. This is based on a given dataset of 120 images.

1.3. OBJECTIVES

The primary tasks for this project, as outlined in the project documentation, were:

1. Prepare a complete data analysis report.
2. Create a classification model for the three disease types.
3. Analyze and report on techniques such as Data Augmentation.
4. Create a Model Comparison Report and suggest a final model.
5. Create a Report on Challenges Faced during the project.

2. DATA ANALYSIS REPORT

2.1. DATASET DESCRIPTION

The dataset provided contains 120 JPG images of disease-infected rice leaves. These are categorized into three classes, with 40 images available for each class. The classes implemented in the project are:

- Bacterial leaf blight
- Brown spot
- Leaf smut

2.2. DATA PREPARATION AND SPLITTING

A challenge was identified in the dataset: the 'Leaf Smut' class contained only 39 images, while the other two had 40. To balance the dataset, one image from the 'Leaf Smut' folder was duplicated before splitting.

The splitfolders library was used to divide the balanced 120-image dataset into training, testing, and validation sets. A ratio of 80% for training, 10% for testing, and 10% for validation was used (ratio=(0.8, 0.1, 0.1)).

This split resulted in the following data distribution:

- **Training Data:** 96 images (3 classes)
- **Testing Data:** 12 images (3 classes)
- **Validation Data:** 12 images (3 classes)

2.3. EXPLORATORY DATA ANALYSIS (EDA)

To visually analyze the data, sample images from the training batch were plotted. This step confirmed that the images were loaded correctly with their corresponding labels (Bacterial leaf blight, Brown spot, Leaf smut), providing a visual understanding of the data the model would be trained on.

3. METHODOLOGY

3.1. DATA PREPROCESSING

All image data was preprocessed using the ImageDataGenerator from Keras. The primary preprocessing step was rescaling the pixel values from the [0, 255] range to the [0, 1] range by applying a rescale = (1./255) factor. This normalization step is crucial for efficient model training.

3.2. DATA AUGMENTATION

Given the small dataset size (96 training images), data augmentation was employed to prevent overfitting and create a more robust model. Augmentation was applied only to the training data using ImageDataGenerator.

The following augmentation techniques were used:

- rotation_range = 40
- width_shift_range = 0.2
- height_shift_range = 0.2
- shear_range = 0.2
- zoom_range = 0.2
- horizontal_flip = True

This process artificially expands the training dataset by creating modified versions of the existing images, helping the model generalize better to new, unseen data.

3.3. MODEL ARCHITECTURE

A Convolutional Neural Network (CNN) was constructed using the Keras Sequential API. The architecture is as follows:

1. **Conv2D Layer:** 32 filters, (3,3) kernel, 'relu' activation, input_shape=(224, 224, 3)
2. **MaxPooling2D Layer:** (2,2) pool size
3. **Conv2D Layer:** 64 filters, (3,3) kernel, 'relu' activation
4. **MaxPooling2D Layer:** (2,2) pool size
5. **Conv2D Layer:** 128 filters, (3,3) kernel, 'relu' activation
6. **MaxPooling2D Layer:** (2,2) pool size
7. **Conv2D Layer:** 256 filters, (3,3) kernel, 'relu' activation
8. **MaxPooling2D Layer:** (2,2) pool size

9. **Flatten Layer:** To convert the 2D feature maps into a 1D vector
10. **Dropout:** 20% neurons turned off
11. **Dense Layer:** 100 units, 'relu' activation
12. **Dropout:** 20% neurons turned off
13. **Dense Layer (Output):** 3 units (for 3 classes), 'softmax' activation

The model was compiled using the adam optimizer and categorical_crossentropy loss function, tracking 'accuracy' as the primary metric.

4. Results and Evaluation

4.1. MODEL TRAINING

The model was trained for **87 epochs** using a **batch size of 16**. The training process utilized the augmented training data, validating against the separate validation set to monitor performance and check for overfitting. The model stopped training at the 87th epoch using the early stopping mechanism. The learning rate was also adjusted during training. Out of a total of 100 epochs, it stopped at epoch 87 because the loss function had not improved for the last 15 epochs, i.e., the **Patience Value was 15**. Finally the best weight from **72th** epoch is selected finally.

4.2. MODEL PERFORMANCE

The model's performance was evaluated by plotting the training and validation accuracy/loss over the 20 epochs and by assessing the final accuracy on the unseen test set.

- **Training Phase Accuracy:** During training phase model achieved a Accuracy of **92% on Training data** and **83% on Validation Data**.
- **Final Test Accuracy:** The model achieved a **Test Accuracy of 75.00%** on the 12 test images.
- **Classification Report & Confusion Matrix:** A classification report and confusion matrix were generated. These results provided a detailed breakdown of precision, recall, and f1-score for each of the three disease classes, confirming the model's ability to distinguish between them with reasonable success. The model (model1.h5) was saved for future use.

5. MODEL COMPARISON REPORT

The project notebook (Prediction.ipynb) focused on the development and evaluation of a single CNN model, designated as “model”. No other models were trained or compared within the notebook.

Therefore, this report is based on the performance of model. With a final test accuracy of 75.00% on a small and challenging dataset, model demonstrates a strong capability for this classification task.

Recommendation: Based on the provided analysis, “**model”** is the suggested model for production.

6. REPORT ON CHALLENGES FACED

Several challenges were identified and addressed during this project:

1. **Small Dataset Size:** The primary challenge was the limited dataset of only 120 images (96 for training). This small size increases the risk of overfitting.
    - **Solution:** Data augmentation (rotation, shifting, zoom, etc.) was applied to the training set to artificially increase its size and diversity.
2. **Unbalanced Classes:** Upon initial inspection, the 'Leaf Smut' class had only 39 images, while the other classes had 40.
    - **Solution:** The dataset was manually balanced by duplicating one image in the 'Leaf Smut' class. This ensured all three classes had 40 images before the 80/10/10 split was performed.
3. **Tuning Data Augmentation:** Finding the correct level of augmentation was a challenge. In the initial stages, overly aggressive augmentation was used, using other image augmenting librarys. This ruined the images by introducing excessive noise and causing important features to be lost.
    - **Solution:** This aggressive augmentation resulted in the model's loss function sky-rocketing and accuracy lowering. The parameters were subsequently re-tuned to the more moderate settings (e.g., `rotation_range=40`, `zoom_range=0.2`) detailed in the methodology, which provided a good balance between variety and feature preservation.

7. CONCLUSION

This project successfully addressed the problem of classifying rice leaf diseases. A Convolutional Neural Network was built, trained, and evaluated. By addressing challenges such as small dataset size and class imbalance through data augmentation and preprocessing, the model achieved a final accuracy of **75.00%** on the test set. This model serves as the recommended solution for the project tasks.