RICE DISEASE

Detection

Presentated by Group 6

12 May 2025

Lecturer: NHIM Malai

Content table

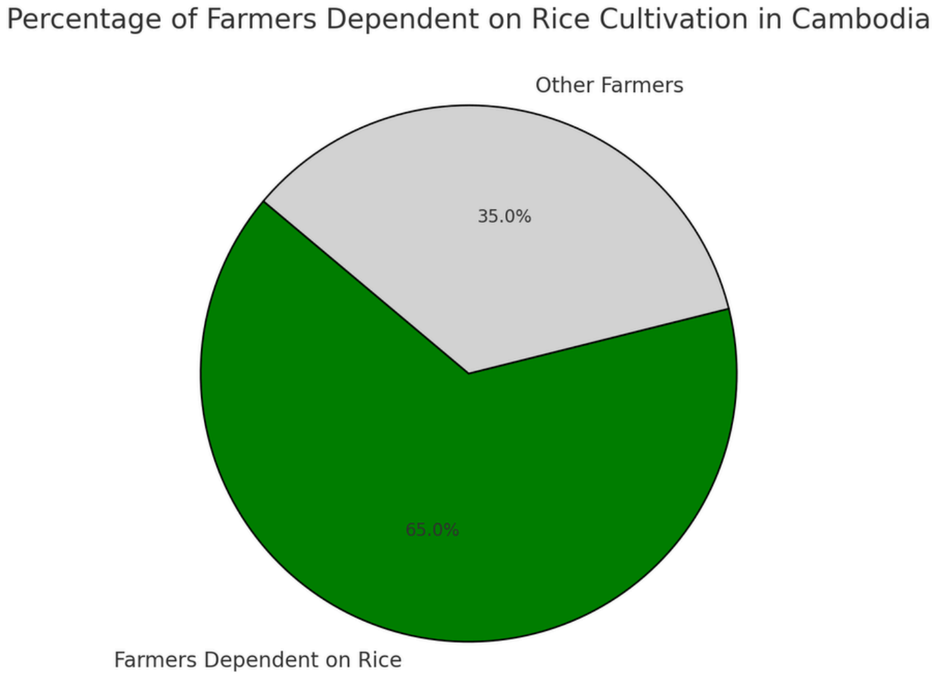
INTRODUCTION

What is Rice Disease Detection?

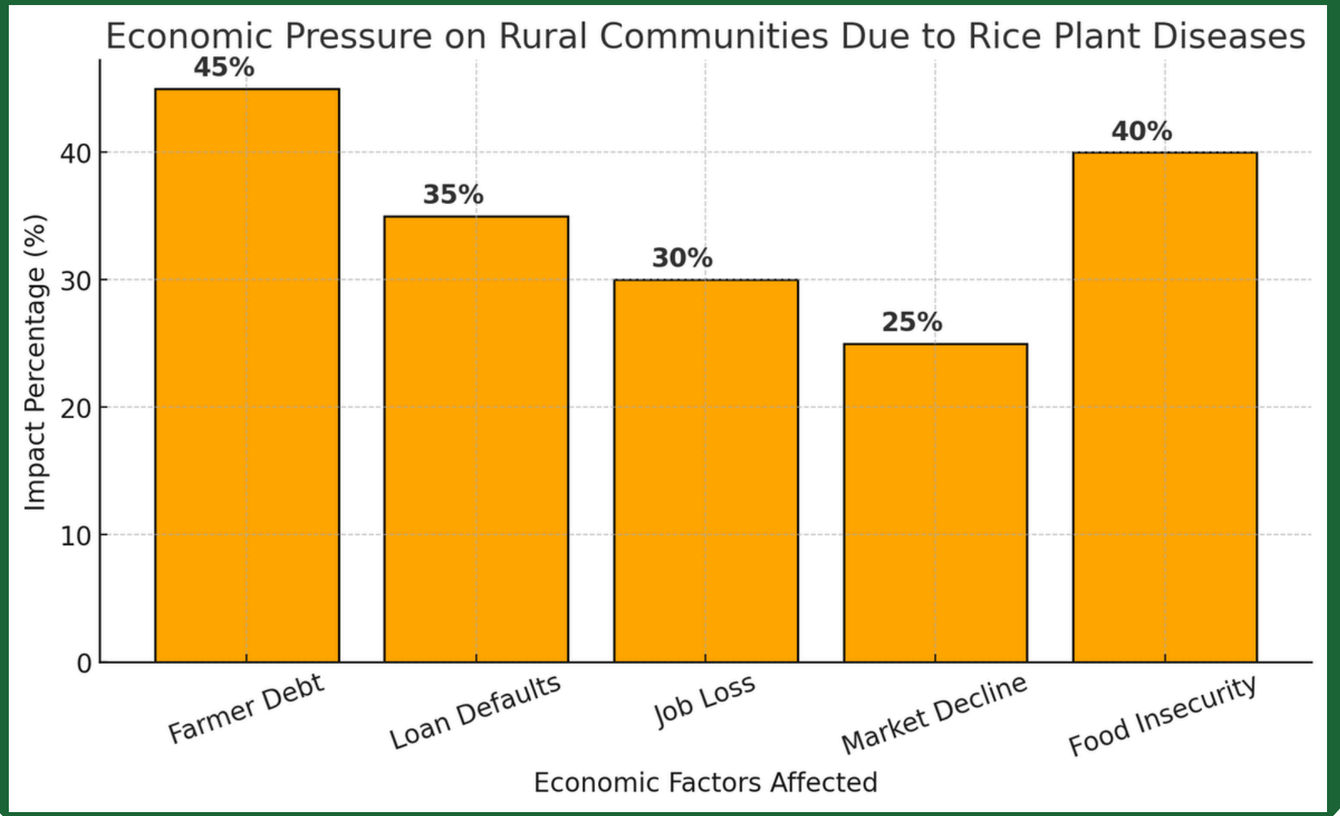
Rice disease detection identifies illnesses in rice plants, crucial for maintaining crop yield and quality since rice is a staple food. Traditionally, farmers inspect plants for visible signs of disease, but new technologies like machine learning, drones, and image processing enhance speed and accuracy. These methods quickly analyze large areas, detect disease patterns, and offer early warnings, enabling effective treatment and minimizing losses for farmers.

PROBLEM STATEMENT

Rice disease detection remains a significant challenge for farmers due to several limitations in traditional methods. Some of the key issues include:



World Bank Reports on Cambodia's Agriculture



OBJECTIVES

deep learning

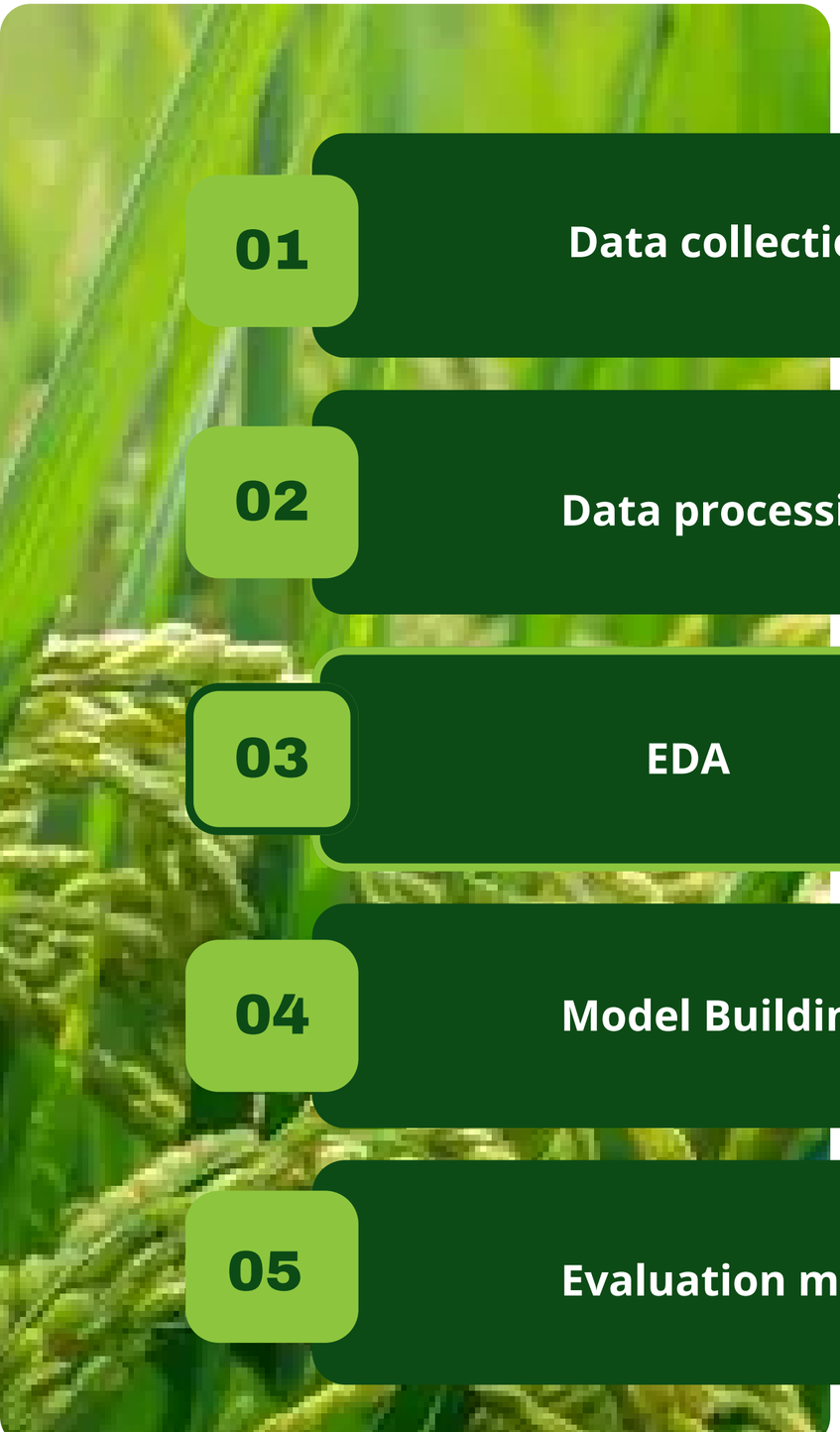
is trained using thousands of rice plant images to accurately classify different diseases, ensuring high precision and reliability in diagnosis.

Real-time diagnosis

helps farmers and researchers detect diseases early, allowing for quick decision-making and timely interventions.

Automated detection reduces dependency on experts

making it accessible to farmers across various regions.



01 Data collection

02 Data processing

03 EDA

04 Model Building

05 Evaluation model

Methodology

Data Collection

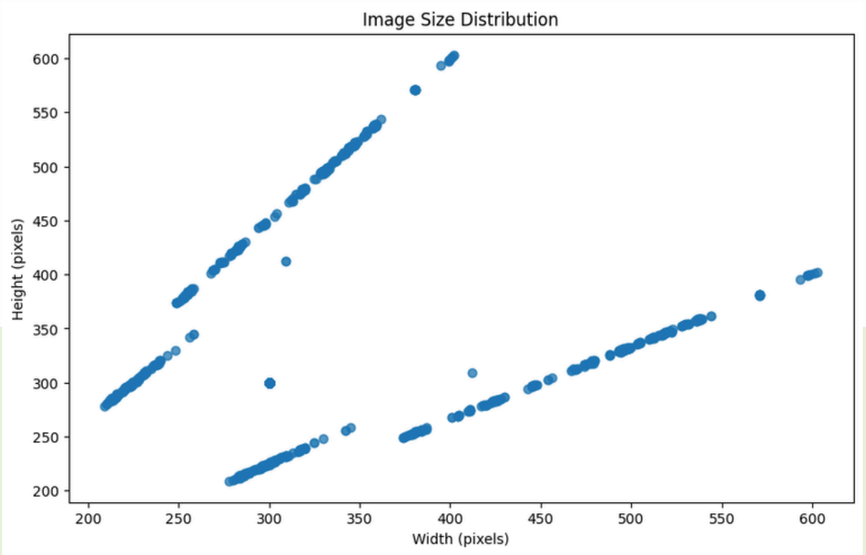
Dataset retrieved from kaggle

Features: There are 4 features regarding to the 4 disease of rice that we are going to detect.
and it has 5932 images

Click Here

Data processing

```
===== CHECKING & DELETING ONLY INVALID IMAGES

Checking class: Blast
Results for Blast:
 - Expected: 1440
 - Actual after cleanup: 1440
 - Deleted invalid files: 0
 - All expected images are present and valid.
```



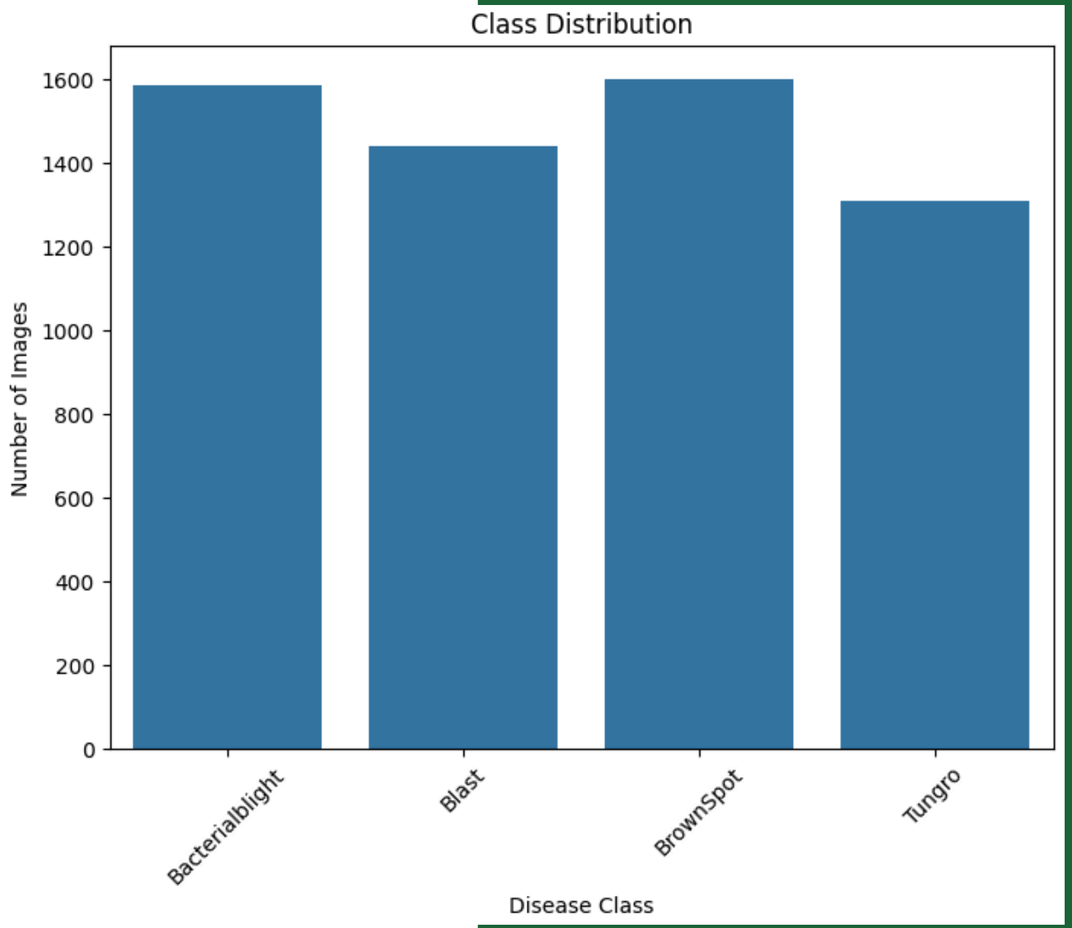
- **Standardizing Image Data:** Resize images, normalize pixel values, and ensure uniform color formats

Check for corrupted and missing or low quality images: can cause errors during training or degrade model performance

Data Argumentation:
Techniques such as rotation, flipping, zooming, and shifting are applied to increase dataset diversity and prevent overfitting.

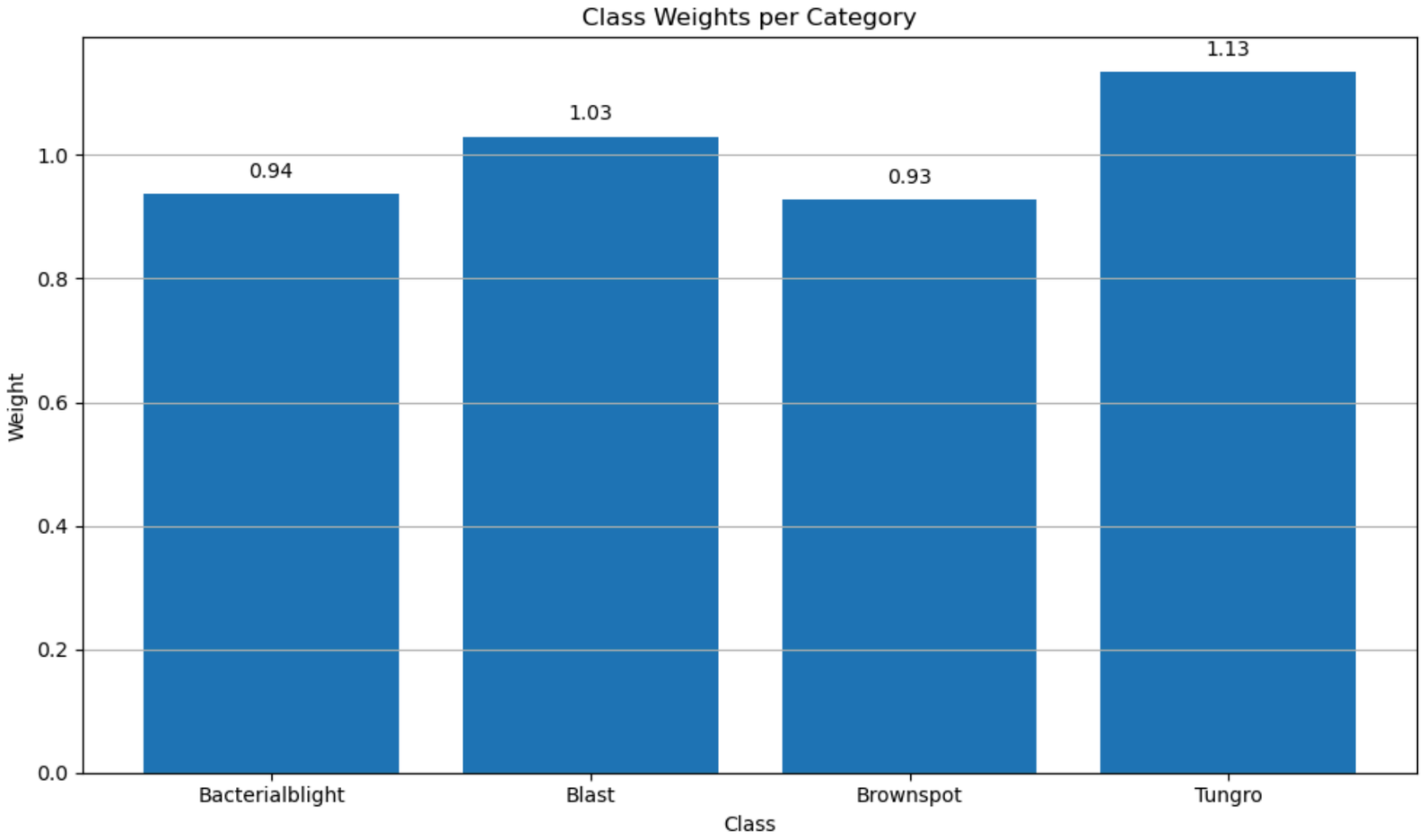
VISUALIZE IMAGE OF EACH CLASS

```
===== ACTUAL IMAGE COUNT PER CLASS =====
Class: Bacterialblight | Images: 1584
Class: Blast | Images: 1440
Class: Brownspot | Images: 1600
Class: Tungro | Images: 1308
Classes in Dataset: ['Bacterialblight', 'Blast', 'Brownspot', 'Tungro']
Total Batches: 186
```



Visualize Sample Images

*VISUALIZE IMAGE OF CLASS WEIGHT



Model Building

The heart of the project

We developed four distinct models : **DenseNet121, MobileNetV2, VGG16, and a custom CNN**.

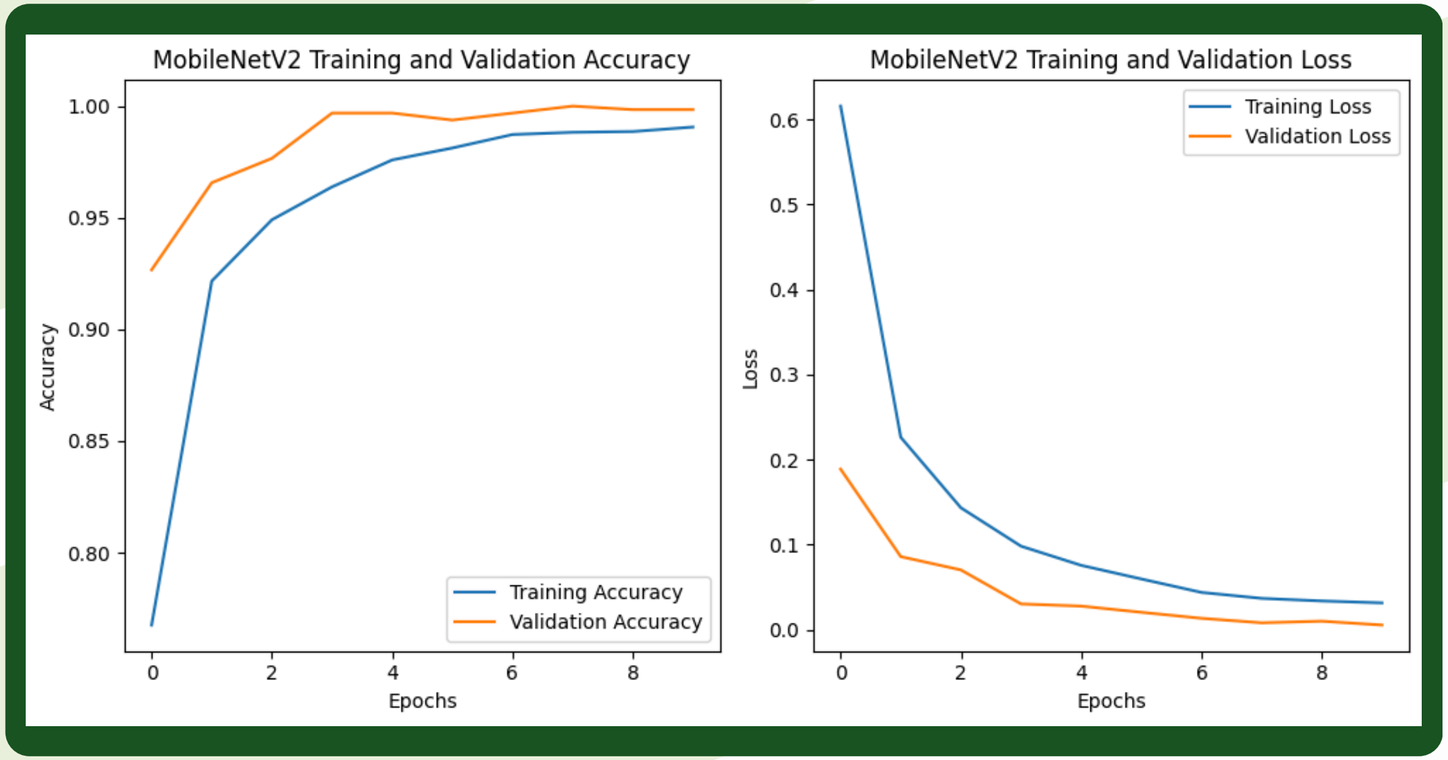
Why four models?
We wanted to compare pre-trained models, which leverage existing knowledge, against a custom model I built to see which performs best for rice disease classification.

Model Architectures

1

MobileNetV2

```python
# MobileNetV2 Model (Transfer Learning)
base_model_mobilenetv2 = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
base_model_mobilenetv2.trainable = False  # Freeze the base model

# Build the custom model
mobilenetv2_model = Sequential([
    base_model_mobilenetv2,
    GlobalAveragePooling2D(),  # Reduce the output to a fixed-size vector
    Dense(256, activation='relu'),
    Dropout(0.7),  # Regularization to prevent overfitting
    Dense(4, activation='softmax')  # 10 classes for multi-class classification
])
```

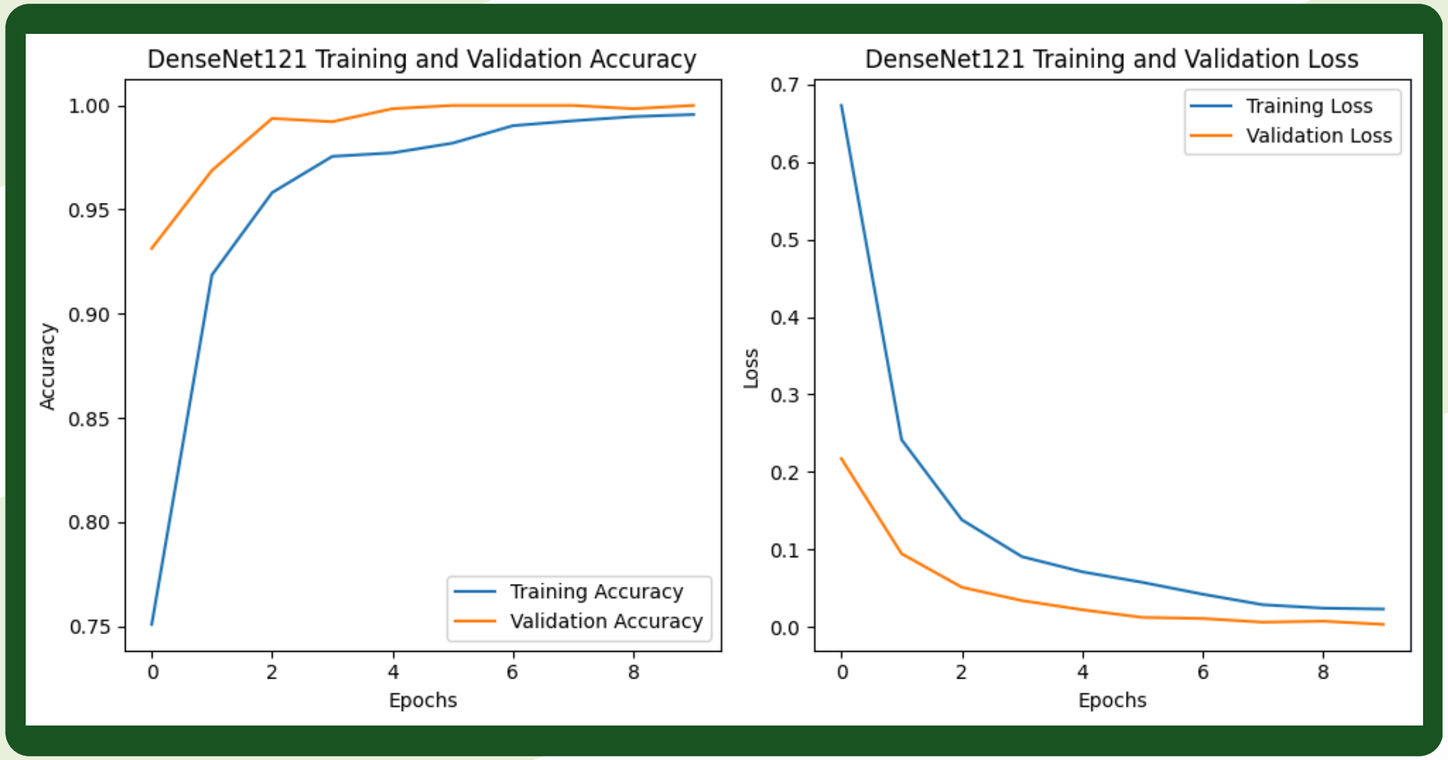


Model Architectures

2

Densenet121

```python
# DenseNet121 Model (Transfer Learning)
base_model_densenet = DenseNet121(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
base_model_densenet.trainable = False  # Freeze base model

densenet_model = Sequential([
    base_model_densenet,
    GlobalAveragePooling2D(),
    Dense(256, activation='relu'),
    Dropout(0.7),
    Dense(4, activation='softmax')  # Assuming 4 classes for the dataset
])
```

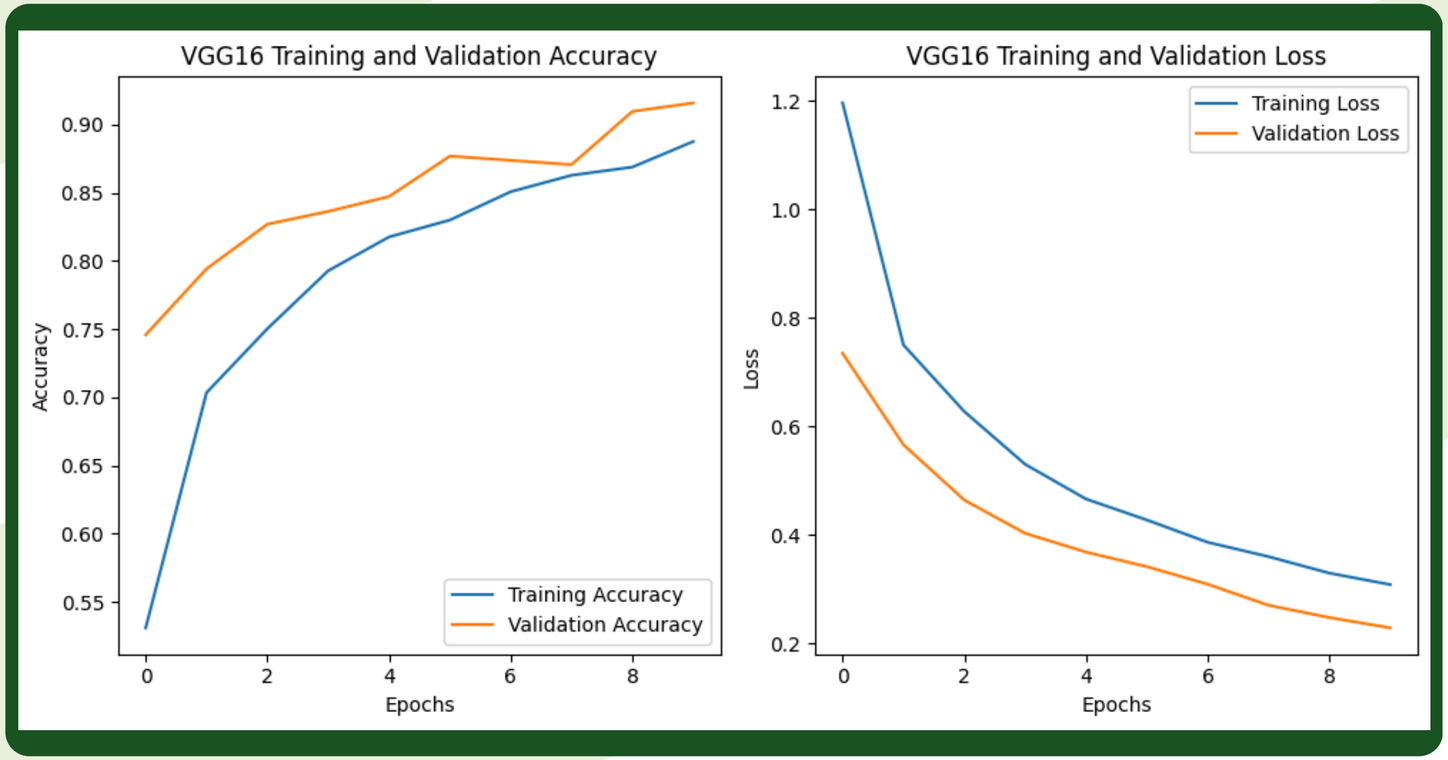


Model Architectures

3

VGG16

```
# VGG16 Model (Transfer Learning)
base_model_vgg16 = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
base_model_vgg16.trainable = False  # Freeze base model

vgg16_model = Sequential([
    base_model_vgg16,
    GlobalAveragePooling2D(),
    Dense(256, activation='relu'),
    Dropout(0.7),
    Dense(10, activation='softmax')  # Assuming 10 classes
])
```

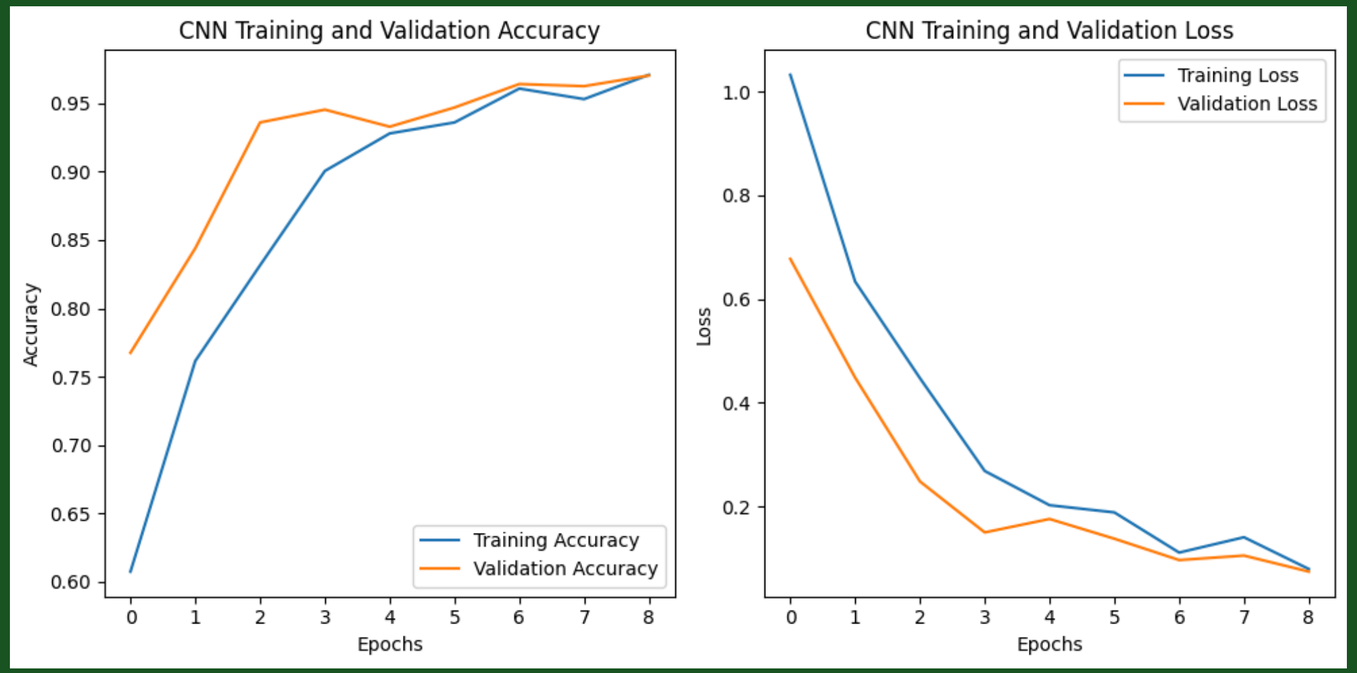


Model Architectures

4

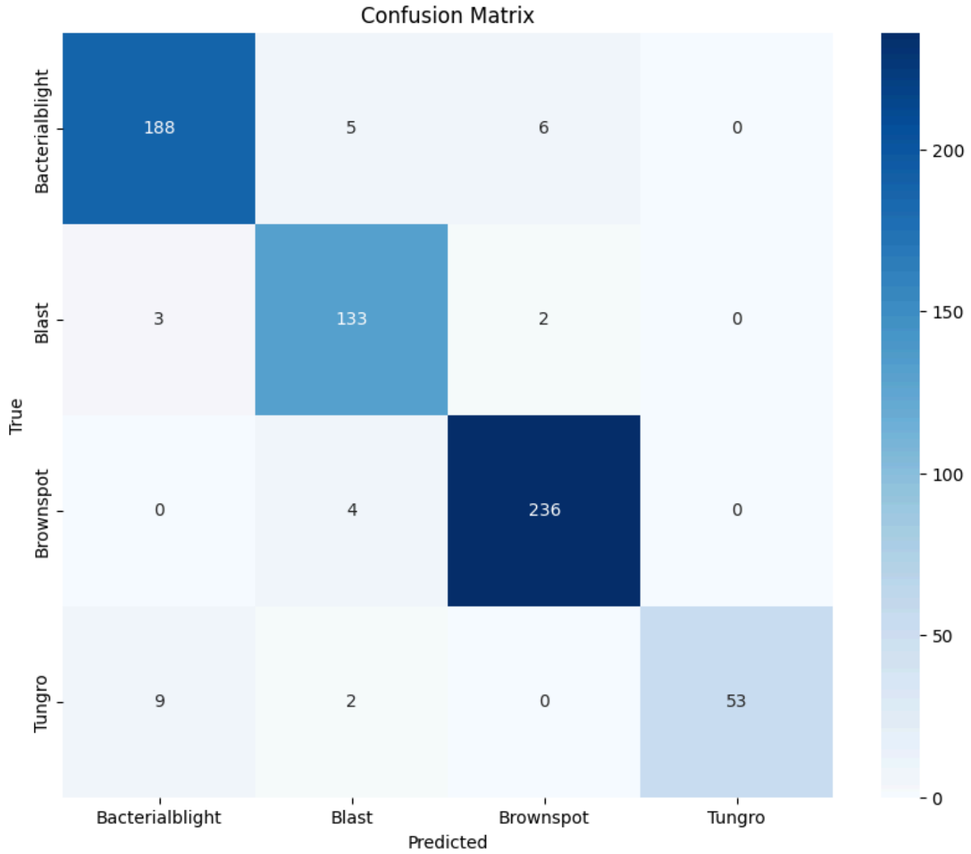
Custom CNN

```
# Define CNN model
cnn_model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    MaxPooling2D(pool_size=(2, 2)),

    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),

    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(pool_size=(2, 2)),

    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.7),
    Dense(4, activation='softmax')  # 4 classes
])
```



Model Evaluation

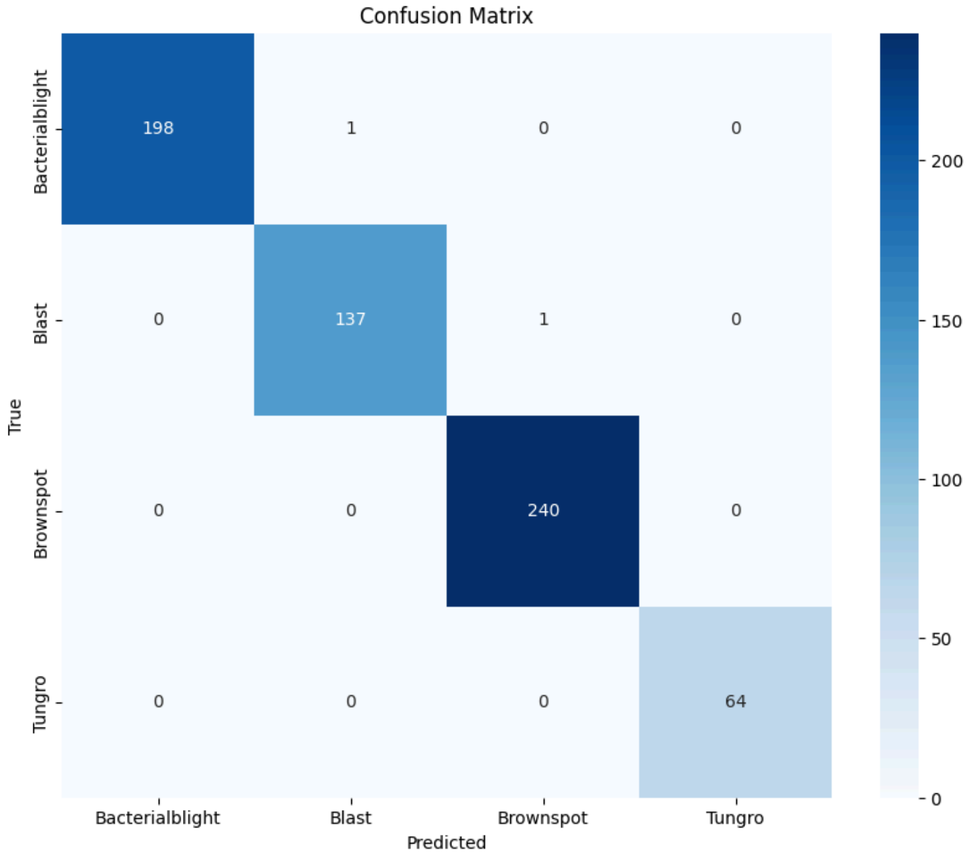
Model 1: CNN



Accuracy= 0.95 = 95%
Accuracy Loss= 0.13 = 13%

	precision	recall	f1-score	support
Bacterialblight	**0.94**	**0.96**	**0.94**	**199**
Blast	**0.92**	**0.83**	**0.94**	**138**
Brownspot	**0.97**	**0.90**	**0.98**	**240**
Tungro	**1.00**	**1.00**	**0.91**	**64**
Accuracy			**0.95**	**641**
macro avg	**0.96**	**0.93**	**0.94**	**641**
weighted avg	**0.95**	**0.95**	**0.95**	**641**

Model 2: Mobilenetv2

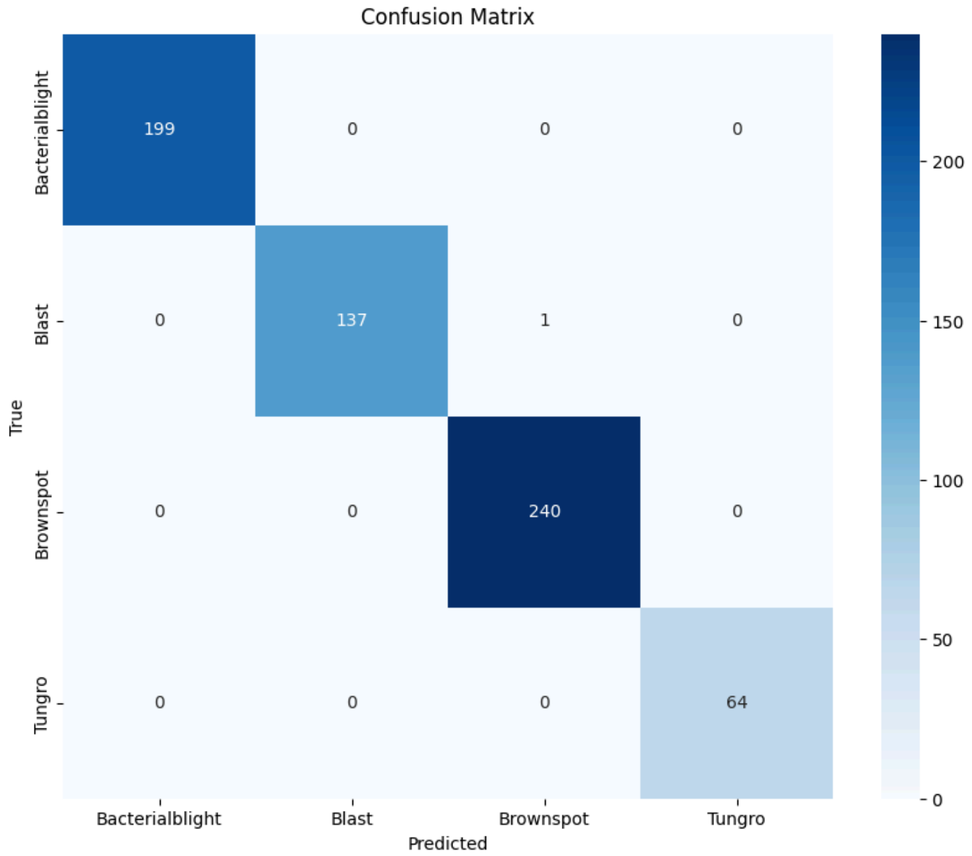


	precision	recall	f1-score	support
Bacterialblight	1.00	1.00	1.00	199
Blast	1.00	0.99	1.00	138
Brownspot	1.00	1.00	1.00	240
Tungro	1.00	1.00	1.00	64
Accuracy			1.00	641
macro avg	1.00	1.00	1.00	641
weighted avg	1.00	1.00	1.00	641

Accuracy= 1.00 = 100%

Accuracy Loss= 0.0128 = 1.28%

Model 3: Densenet

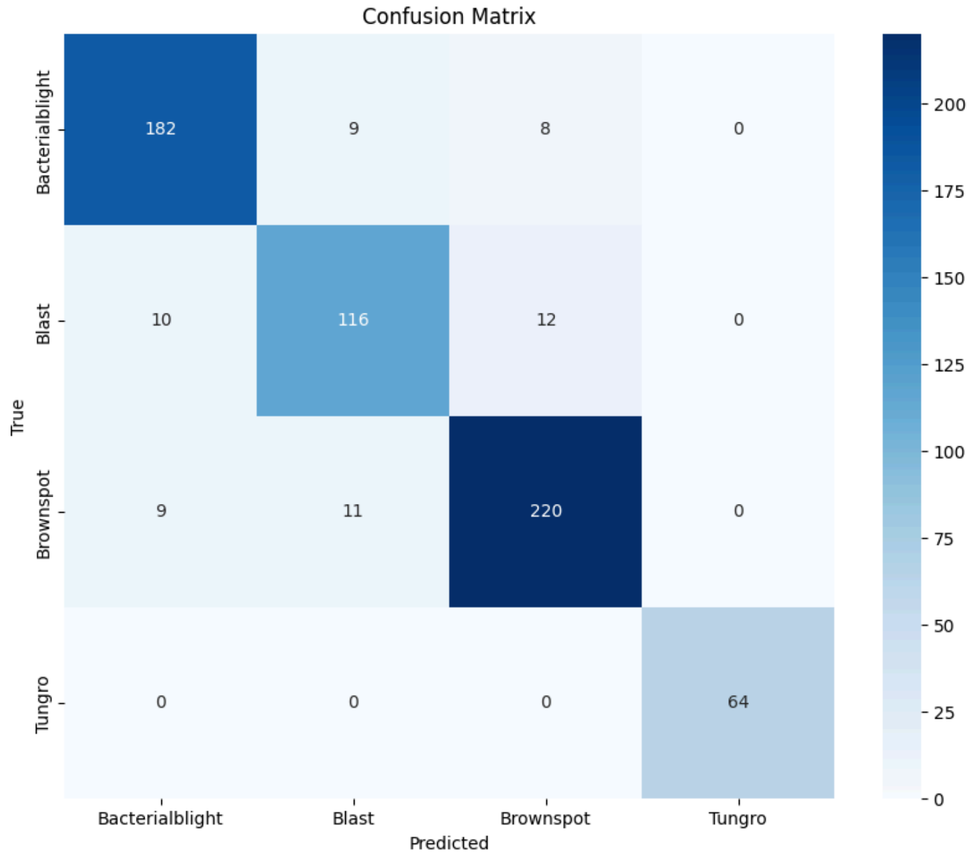


Accuracy= 1.00 = 100%

Accuracy Loss= 0.0050 = 0.5%

	precision	recall	f1-score	support
Bacterialblight	**1.00**	**1.00**	**1.00**	**199**
Blast	**1.00**	**0.99**	**1.00**	**138**
Brownspot	1.00	**1.00**	**1.00**	**240**
Tungro	**1.00**	**1.00**	**1.00**	**64**
Accuracy			**1.00**	**641**
macro avg	**1.00**	**1.00**	**1.00**	**641**
weighted avg	**1.00**	**1.00**	**1.00**	**641**

Model 4: VGG16



	precision	recall	f1-score	support
Bacterialblight	0.95	0.92	0.94	199
Blast	0.85	0.93	0.87	138
Brownspot	0.94	0.94	0.94	240
Tungro	1.00	1.00	1.00	64
Accuracy			0.93	641
macro avg	0.94	0.94	0.94	641
weighted avg	0.93	0.93	0.93	641

Accuracy= 0.91 = 91%

Accuracy Loss= 0.2497 = 24.97%

Models comparison

MODEL	ACURRACY
MobileNetV2	100%
VGG16	93%
DenseNet	100%
CNN	95%

Conclusion

The project on rice disease detection successfully demonstrates the potential of advanced technologies, such as machine learning and image processing, in accurately identifying and classifying rice plant diseases. By leveraging datasets of rice plant images and training models to recognize disease patterns, the system achieves high precision in early detection, enabling timely intervention to mitigate crop losses.

THANK YOU

Phone

+123-456-7890

Address

Phnom Penh

Webs

reallygreatsite.com