**BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING**

# Improving Rice Leaf Disease Identification with Object Detection and Image Enhancement

**Rafi Hassan Chowdhury**
**190041234**
**Faria Ahmed**
**190041218**
**Tasfia Tasneem Annesha**
**190041220**

**Department of Computer Science and Engineering**
Islamic University of Technology
June, 2024

# Improving Rice Leaf Disease Identification with Object Detection and Image Enhancement

**Rafi Hassan Chowdhury**
**190041234**
**Faria Ahmed**
**190041218**
**Tasfia Tasneem Annesha**
**190041220**

**Department of Computer Science and Engineering**
Islamic University of Technology
June, 2024

# Declaration of Candidate

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by **Rafi Hassan Chowdhury**, **Faria Ahmed**, and **Tasfia Tasneem Annesha** under the supervision of **Tareque Mohmud Chowdhury**, Assistant Professor, Department of Computer Science and Engineering and co-supervision of **Njayou Youssouf**, Lecturer, Department of Computer Science and Engineering and **Sabrina Islam**, Lecturer, Department of Computer Science and Engineering, Islamic University of Technology, Dhaka, Bangladesh. It is also declared that neither this thesis nor any part of it has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of references is given.

---

**Tareque Mohmud Chowdhury**
Assistant Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: June 04, 2024

---

**Njayou Youssouf**
Lecturer
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: June 04, 2024

---

**Sabrina Islam**
Lecturer
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: June 04, 2024

---

**Rafi Hassan Chowdhury**
Student ID: 190041234
Date: June 04, 2024

---

**Faria Ahmed**
Student ID: 190041218
Date: June 04, 2024

---

**Tasfia Tasneem Annesha**
Student ID: 190041220
Date: June 04, 2024

*Dedicated to the hardworking farmers of Bangladesh. Their tireless efforts, resilience, and unwavering dedication to feeding the people inspire us every day. We hope that this research contributes in some small way to easing their burdens and improving their livelihoods. Thanks to them for their invaluable contributions to society.*

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **CNN** | Convolutional Neural Network |
| **CNet** | Classifier Network |
| **ViT** | Vision Transformer |
| **EffVIT** | Efficient Vision Transformer |

# Acknowledgement

First and foremost, we would like to express our deepest gratitude to our supervisor, Tareque Mohmud Chowdhury, for his invaluable guidance, support, and encouragement throughout this research journey. His expertise and insightful suggestions on a larger scale have significantly shaped the direction and quality of this work.

We are also profoundly grateful to our co-supervisors, Njayou Youssouf and Sabrina Islam, for their continuous involvement and timely advice. Their hands-on assistance in figuring out various challenges and dedication throughout the research period has been crucial to completing this project.

Thanks to the Bangladesh Rice Research Institute (BRRI) for helping us with the required information and assistance.

Thanks to our friends and faculties for their support, encouragement, and countless discussions that sparked new ideas and perspectives. Their moral support has been a constant source of motivation.

Finally, to our families, especially our parents, their unwavering love, patience, and encouragement have been our anchors throughout this journey. This achievement would not have been possible without their constant support and belief in us.

# Abstract

Highly accurate rice leaf disease identification using lighter models can ensure food security globally and also less crop loss by maximizing the rate of disease identification in proper time. Our thesis focus is on the process where the quality of the image is enhanced and the diseased part is detected for predicting the disease in the output. Image enhancement refers to the process where noisy, low-contrast, and low-quality image is taken care of. The object detection mechanism detects which part of the image contains disease. The identification process specifies the disease from the given input. The currently available disease detection model seems to have limitations, such as - high-quality training and testing images, requiring high computational power, images with biased backgrounds, etc. Our proposed technique works on these limitations by including an image enhancement technique, Lite SR-GAN, that gives a better result even if we train our models on low-quality images. To make the computation less complex we have used lightweight architectures in the different phases (image enhancement, detection, disease identification) of the model. Next, to address the issue of biased images, we have used an object detection technique, YOLO-s, to detect the diseased part of the image. In the final stage, we introduce a lightweight model, EfficientViT that, with the help of previous layers, works equally fine with pictures of all qualities and identifies the disease using less computation power with an accuracy of 97.615%, inference time 33.99ms per image, parameter count 4.56M, model size 6.95MB and flop count 33.56B. .

# Chapter 1

# Introduction

## 1.1 Overview

Image enhancement leads to more perfection in detecting disease from a low-quality image. Detecting the diseased part correctly leads to a greater chance of identifying the disease with better accuracy.

### 1.1.1 Image Enhancement

Image enhancement refers to the different techniques for improving the visual quality of the images. Various methods can be applied to make the images look better than their original form. [33] Using the image quality enhancement techniques we can upgrade the image quality so that all the details are easier to locate. It will help us eliminate the blurriness and the breaking effect when we try to zoom in.

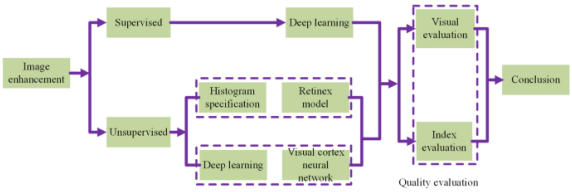


**Figure 1.1:** Different image enhancement techniques[42]

In the particular scenario where we are working with rice leaf disease enhancement, our goal is to improve the image quality so that locating the disease part from the given image becomes easier.

### 1.1.2 Object Detection

Object detection[43] is a computer vision[37] task that helps locate and identify an object within the image. This gets a boundary around each object and puts labels on them.

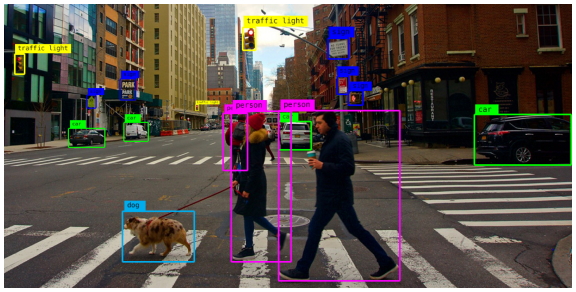


**Figure 1.2:** Object detection[27]

Object detection in our work will help us locate the diseased part from an image and help us in the binary classification of whether any disease exists in the image or not. From the enhanced image this will be easier to detect what we need from an image with a noisy background.

### 1.1.3 Identification

From the Computer vision[40] point of view, identification means relating and remembering different entities based on their characteristics. It remembers similar characteristics to be associated with the same class and unrelated characters to be associated with different classes

Disease identification[10] refers to the method that helps us identify different diseases. After locating the diseased part from the image we need to train the model with an identification model to identify which class it belongs to.

### 1.1.4 Real-World Applications

The practical relevance of this research lies in its direct applications in agriculture, particularly in aiding farmers with disease management in crops. Some of the key real-world applications include:

- **Crop Disease Management:** Farmers can use their mobile phones to quickly and accurately identify diseases in their crops, enabling them to take immediate corrective actions. This reduces crop losses and increases yield, directly impacting their income and livelihood.

- **Extension Services:** Agricultural extension workers can utilize the disease identification model to assist multiple farmers across different regions. This improves the efficiency and reach of extension services, ensuring timely advice and support.
- **Precision Agriculture:** The integration of this model into precision agriculture practices can enhance decision-making processes. Farmers can monitor the health of their crops more closely, applying treatments precisely when and where needed, thereby optimizing resource use and reducing costs.
- **Training and Education:** The model can serve as an educational tool for farmers, helping them learn to recognize different crop diseases and understand best practices for disease management. This knowledge transfer can lead to more informed farming communities.
- **Supply Chain and Quality Control:** Agribusinesses involved in supply chain management can use the model to ensure the quality of produce. Early detection of diseases can prevent the spread of infected crops, maintaining the quality and safety of agricultural products.
- **Government and Policy Implementation:** Government agencies can leverage this technology to gather data on crop disease prevalence and spread, aiding in the formulation of targeted agricultural policies and intervention programs.
- **Remote and Underserved Areas:** In remote and underserved regions where access to agricultural experts and advanced diagnostic tools is limited, this mobile based solution can provide critical support to farmers, ensuring they are not left behind in technological advancements.

By facilitating timely and accurate crop disease detection, this research has the potential to transform agricultural practices, improve food security, and enhance the economic well-being of farmers worldwide.

## 1.2 Motivations and Scope

In Bangladesh, rice is the basic diet for around 135 million people. It supplies almost two-thirds of the nation's entire calorie supply, half of the average person's total protein consumption and nearly 48% of rural employment. In Bangladesh, the rice industry accounts for one-sixth of the country's total income and half of the agricultural GDP. Rice is cultivated on almost 75% of the total farmed land and more than 80% of the total irrigated area. As a result, rice is essential to Bangladeshi people's way

of life. Thirteen million agricultural households in the nation cultivate rice, almost all of them. Over the past three decades, the amount of land used to cultivate rice has stayed relatively constant at 10.5 million hectares.[18] The primary motivation for this research is the substantial losses village farmers face each year due to various crop diseases. These losses not only impact the farmers' livelihood but also affect food security and the agricultural economy. Timely identification and management of crop diseases are crucial to mitigate these losses. If farmers can detect crop diseases using their mobile phones, they can address the issues promptly, thereby minimizing damage and preserving crop yield. From the previous works, we came across the limitations of this research field. The major ones involve the training and testing dataset. As the models are being trained on high-quality images with biased backgrounds, the accuracy rate seems to be very high, however, they don't perform very well with low-quality images. Another limitation of the images is they seemed to have biased backgrounds, not the real field background. Another limitation of the previous works is the computation power it needs.

The scope of this research encompasses the development of a disease identification model with a lightweight architecture suitable for deployment on mobile devices. This model aims to allow farmers to take care of their crops from their homes. The system is designed to handle input images of any resolution, enhancing the images before processing them through object detection algorithms to identify diseases.

## 1.3 Problem Statement

Farmers in rural areas often face significant challenges due to the lack of accessible, efficient, and accurate tools for the timely identification and management of crop diseases. This deficiency leads to substantial crop losses and consequent economic hardship. Without proper resources, diseases can spread unchecked, severely impacting crop yield and quality. The economic strain from these losses can be devastating for farmers, affecting their livelihood and the stability of their communities.
In short,

*"Farmers in rural areas lack accessible, efficient, and accurate tools for timely identification and management of crop diseases, leading to significant crop losses and economic hardship."*

Addressing these issues by providing user-friendly, quick, and precise disease detection tools can greatly enhance disease management practices, reduce crop losses, and improve the economic well-being of rural farmers. We have divided our aims and

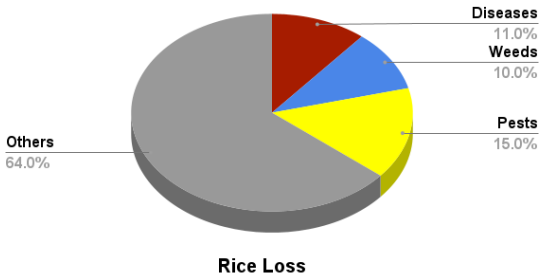


**Figure 1.3:** Different Rice Loss Causes[25]

objectives into two phases. Both phases are discussed in the following part.

### 1.3.1 Aim-1

- **Rice leaf image enhancement and object detection.**

#### Objective

- Determining the performance on a low-quality dataset.
- Use image enhancement method for better object detection.
- Evaluating the effects of data enhancement technique on the object detection performance.
- Ensuring mobile compatibility.

### 1.3.2 Aim-2

- **Identify the input image in 10 different disease classes.**

#### Objective

- Identifying diseases with high accuracy using light architecture models.
- Developing an efficient system that combines image enhancement and disease detection techniques, specifically tailored for paddy crop disease identification.

## 1.4 Research Challenges

Researchers in the domain of crop disease identification using mobile devices face several significant challenges:

- **Models Trained and Tested on High-Resolution Images:** Most of the existing models are trained and tested on high-resolution images, which ensures high accuracy in output. However, in reality, there is a high possibility of getting lower-resolution images, leading to reduced accuracy and effectiveness of the models in real-world scenarios.
- **Limited Access to High-Resolution Devices for Farmers:** Farmers in rural areas may not have access to high-resolution imaging devices. Their mobile phones often have basic cameras that do not capture the detailed images required by advanced disease detection models, limiting the models' applicability.
- **Low Computational Power of Mobile Devices:** Mobile devices which are affordable to rural farmers, typically have limited computational power. This poses a significant challenge for running complex machine learning models that require substantial processing power, leading to slower performance and potentially less accurate results.
- **Challenges in Data Collection:** Collecting a diverse and representative dataset of diseased crop images from various locations and under different conditions is difficult. This is compounded by logistical issues and the need for expert validation of the disease labels in the collected images.
- **Inadequate Size of Available Dataset:**The available datasets for training crop disease identification models are not large enough to cover the wide variety of diseases and the variability in how these diseases appear on different crops. This inadequacy can result in models that do not generalize well to new, unseen data.
- **Varied Environmental Conditions:** Environmental conditions such as lighting, weather, and background can vary significantly and affect the quality of the images captured. The available models & datasets have not considered this scenario. As no data is available for this consideration researcher finds it hard to train their model for this.
- **Biased Images in the Available Dataset:** Many datasets contain biased images with biased background. These give high accuracy for the provided dataset but fail in practical, on-field conditions where background differ significantly.
- **Lack of Integration of Image Quality Enhancement Techniques:** There is often a lack of integration of image quality enhancement techniques in existing models. Enhancing the quality of low-resolution or poorly lit images before

feeding them into the disease identification model significantly improves accuracy.

To address these challenges, we need to develop innovative solutions that are tailored to the practical issues faced by rural farmers. This includes creating lightweight models that can run on low-power devices, incorporating image enhancement techniques, and ensuring that the models are trained on diverse and representative datasets. By overcoming these hurdles, researchers can create more effective tools for crop disease identification, ultimately benefiting farmers and improving agricultural productivity.

## 1.5 Contribution

To improve current state of the research area of rice leaf disease identification we have introduced our model with following contributions -

- **Development of a Lightweight Disease Identification Model:** We created a highly efficient model that can run on low-power mobile devices, making advanced crop disease detection technology accessible to rural farmers.
- **Image Enhancement Techniques for Low-Resolution Inputs:** We integrated image enhancement methods to pre-process and improve the quality of low-resolution images, ensuring accurate disease detection regardless of the input image quality.
- **Robust Model Performance Under Varied Environmental Conditions:** Our image enhancement model is designed to handle a wide range of environmental variations, such as differing lighting and background conditions, to provide consistent and reliable disease identification in real-world scenarios.
- **Comprehensive Dataset Collection and Augmentation:** We compiled a diverse and representative dataset of diseased crop images and applied data augmentation techniques to address the inadequacy in available datasets.
- **Biased image correction:** We have used an object detection method using bounding boxes to specifically detect the diseased part in the image. This helps us deal with the biased background issue of the existing dataset.

The importance of introducing this model & how it advances the field are as discussed below-

- **Increased Accessibility for Farmers:** By developing a lightweight model that can operate on commonly available mobile devices, our research makes ad-

vanced crop disease detection tools accessible to a broader range of farmers, particularly those in rural areas.

- **Improved Accuracy and Reliability:** The integration of image enhancement and object detection techniques and the robust handling of environmental variations enhance the model's accuracy and reliability, ensuring farmers receive precise disease identification and timely intervention.
- **Enhanced Data Representatives:** The enhancement and object detection of a diverse dataset contribute to the creation of more generalized models, reducing the impact of bias and improving performance across different crop types and conditions.
- **Empowerment of Rural Farmers:** Our research empowers farmers by providing them with the tools to identify and manage crop diseases independently, thereby improving their agricultural productivity and economic stability.

## 1.6 Organization of the Thesis

We have organized the remaining part of the thesis in the following manner –

In Chapter 2 we will talk about some existing works on image identification and image enhancement. We will have an in-depth discussion on the literature we found to be relevant to our work.

In Chapter 3 we are going to give a detailed explanation about our proposed methodology.

In Chapter 4 we will discuss the datasets we are working on and the experiments we have already conducted. Here we will be analyzing all the results and make comparisons with existing works as well as previous results.

Finally, in Chapter 5 we will conclude our discussion with a conclusion from the previous discussions and also discuss our future goals and limitations.

# Chapter 2

# Related Works

Rice cultivation and maintenance are vital around the world, particularly in Asia, where rice is a staple meal. Rice supplies essential protein and carbohydrates to nearly 170 million people in Bangladesh, making it their main source of food. Rice production has a considerable economic impact on Bangladesh's agricultural GDP, contributing 4.4% and accounting for a large portion of national income. Nearly all of Bangladesh's 13 million farming households cultivate rice on 10.5 million hectares of land, cementing the country's status as the world's fourth-biggest rice producer. In modern agriculture, traditional methods for detecting diseases in plants have mostly been labor tedious and not that accurate. Consequently, introducing sophisticated technology like Convolutional Neural Networks (CNNs) in the field of agriculture is paramount.CNNs have shown their efficiency in various image classification tasks of which agricultural applications are also inclusive turning to recent studies. They become suitable for feature extraction and classification purposes due to their ability to develop intricate structures of features from nothing else but unprocessed pixel information. Still, advanced feature extraction techniques can enhance the performance of CNNs to a larger extent.

Recent research has shown the utility of advanced machine learning models in the detection and categorization of rice diseases.

## 2.1 Memory-efficient Simple CNN architecture

Rahman et al.[30] introduced Simple CNN, a memory-efficient CNN architecture that achieves good accuracy without any prior training on the ImageNet dataset[6]. The study also introduces a new concept of two-stage training, derived from the concept

of fine-tuning, which enables the Simple CNN architecture to perform well in real-life scenarios.This article next showed a comparison among multiple CNN architectures[3] to recognize rice diseases and pests:

- **VGG16**: A widely-used architecture with a large number of parameters, it achieved high accuracy but suffers from memory inefficiency [12].
- **InceptionV3**: A non-sequential architecture that employs inception blocks for better feature extraction [26].
- **MobileNetv2 and NasNet Mobile**: Designed for memory-efficient applications on mobile devices.
- **SqueezeNet**: A compact architecture with fewer parameters[35].
- **Simple CNN**: A novel two-stage training-based CNN with significantly reduced model size.

In the paper the comparison among architectures indicates the effectiveness of transfer learning and fine-tuning in improving the model's performance, especially with VGG16 and MobileNetv2 showing promising results.The research described in this paper collected a dataset comprising 1426 images of rice diseases and pests from BRRI (Bangladesh Rice Research Institute)[17]. The dataset covers nine classes, including eight classes of diseases and pests and a healthy plant class. The dataset represents various weather conditions and intra-class variations [4]. Despite the progress in rice disease and pest detection using CNNs, there are several limitations:

- Some CNN architectures required a large number of parameters, posing resource challenges.
- The need for large-scale architectures like VGG16, which may not be suitable for mobile applications in resource-constrained environments.
- The challenge of handling images with heterogeneous backgrounds.
- Potential misclassification of minor symptom variations such as variations in symptoms at different disease stages presented classification difficulties.
- Additionally, The study acknowledged dataset bias, where diseases of the same class had the same backgrounds, which may affect model generalization.

## 2.2 DenseNet-based channel attention mechanism

Jiang et al. introduced an attention-based deep dense network in their study [19], which enhances feature extraction for disease identification in rice plants using Squeeze-and-Excitation (SE) blocks. Here the denseNet architecture consists of dense blocks and transition layers which serves as the model's backbone. Furthermore, the study develops an attention technique known as "squeeze-and-excitation" (SE), which enhances crucial features while suppressing less valuable ones. Depth-wise separable convolutions are used to improve parameter utilization and training speed. The

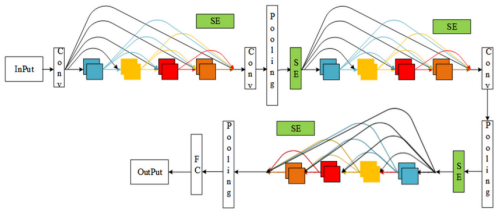


**Figure 2.1:** SE DenseNet Model Architecture

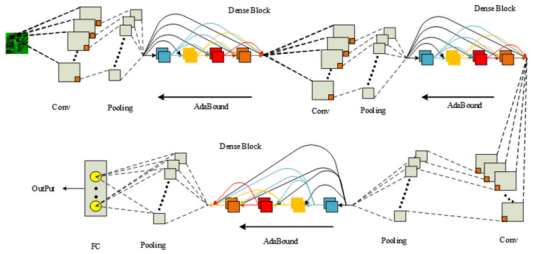


**Figure 2.2:** AB-SE-DenseNet model architecture

proposed method is evaluated on five different datasets of rice disease images: rice blast, blight, brownspot, sheath blight, and tungro.To enhance the dataset, the images were altered to simulate different lighting conditions (sunny and cloudy days), horizontal inversion, viewing angles, and colors to mimic occlusions. This augmentation resulted in a total of 4235 images. The dataset was split in a ratio of 6:2:2 for training, validation, and testing, respectively. The images were preprocessed to a size of $224 \times 224$ pixels, and cross-entropy loss was used as the loss function during model training.

In terms of accuracy, the proposed model outperformed the DenseNet model with the ECA attention module[14], reaching 99.4% versus 99.2%. It also showed faster recognition, taking 3.71 seconds against 8.67 seconds with the ECA attention module. In contrast, when compared to the DenseNet model with the CBAM attention module, the suggested model had slightly lower accuracy (99.4% vs.99.8%). However, it was

recognized substantially faster, taking only 3.71 seconds compared to 29 seconds for the CBAM attention module. The AdaBound algorithm, combined with adaptive optimization, is employed to reduce parameter adjustment time during training. Future research directions suggested by the authors include using deep learning to assess disease severity. The main contribution of this paper is:

- An improved DenseNet-based rice disease identification method.
- Three variants of the SE-DenseNet network with SE modules embedded at different locations.
- The introduction of the AdaBound optimization algorithm.

## 2.3 Two-Stage Learning and YOLOX

In a novel approach, Liu et al.[29] introduced RiceNet, a two-stage machine learning method utilizing YOLOX for object detection followed by a Siamese network for disease classification [2]. In this paper the model combines a segmentation network and an identification network to achieve satisfactory identification accuracy for rice diseases. It was trained and tested on a dataset of 200 rice disease images which were augmented to create a clipped patch dataset containing a total of 626 images. These images were obtained from the Experimental Practice and Demonstration Station of Northeast Agricultural University in Fangzheng county during September 2019. All the images were captured in rice fields and are in RGB color format. The images feature complex backgrounds, which is typical for field-collected data, and it showed promising performance in identifying various types of rice diseases.RiceNet[29] addresses the challenges of limited annotated images and complex background information in rice disease identification through a two-stage method.

In the first stage of the proposed model by Liu et al. YoloX [29] is used for object detection to locate and clip rice disease patches, effectively reducing the complex background information. This process forms a new rice disease patch dataset, which helps in overcoming the limitations of limited annotated images and complex background information.

In the second stage, a Siamese Network is employed for disease identification, achieving high identification accuracy.
This study highlighted high-resolution imagery as a need for model accuracy, indicating a potential performance gap for lower-resolution photos that are more common in real-world circumstances. But this paper focused on just four specific rice diseases,

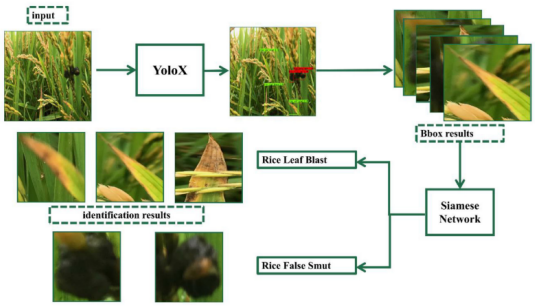


**Figure 2.3:** Processing flow

but some many more diseases and pests affect rice. It would be beneficial to study a wider range of rice diseases and pests to better protect rice crops. Additionally, all the images taken here are high resolution (3024 x 4032 taken by iPhone 7 or Huawei P10) images. The model may not work very well on low-resolution images. The paper talked about a model for identifying rice diseases. But what they didn't do is turn that model into a practical tool that farmers can use easily.

## 2.4 Image Super-Resolution and Segmentation

El-Assiouti et al. [32] have proposed two innovative lightweight architectures, Lite-UNet and Lite-SRGAN, for image super-resolution, segmentation, and localization for plant leaf diseases. These architectures outperformed complex networks while significantly reducing computational complexity and inference time. The dataset used in this research is the Plant Village dataset [1]. Lite-SRGAN achieves comparable quali-

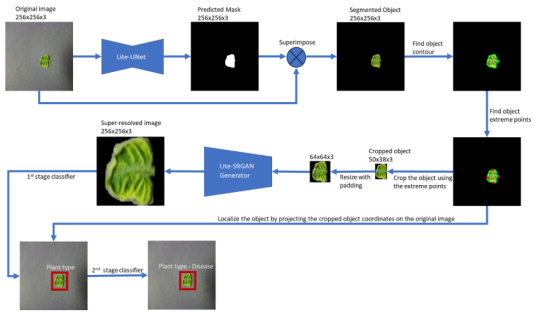


**Figure 2.4:** Processing flow

tative and quantitative results compared to SRGAN, with reductions of 7.5x, 7.8x, and 2.7x in terms of parameters, FLOPs, and inference time when upsampling from 64x64 to 256x256, and 7.1x, 11.2x, and 1.9x when upsampling from 128x128 to 256x256. The advantages of lightweight models in plant disease detection and classification were

highlighted, emphasizing their efficiency in terms of resource utilization, reduced latency and cost, high-speed inference, memory efficiency, and overall superior performance compared to complex networks. These advantages make lightweight models a compelling choice for practical applications in resource-constrained environments. However, the training dataset contains no background noise, which may limit the model's capacity to perform well in noisy real-world circumstances. Furthermore, the dataset consists primarily of close-up photos, which may limit the model's performance when confronted with wider-angle or distant views. Finally, the model's reliance on higher-resolution photos for training and testing may impede its performance when applied to lower-resolution images typically seen in real-world contexts, limiting its adaptability.

## 2.5 Enhancing Rice Crop Management

In this paper[13] a Convolutional Neural Network (CNN) model developed for the onset detection and categorization of rice illnesses by using picture processing with an accuracy as high as 97.9%. It was trained on 2700 disease-rice datasets hence its implementation in an android app that tracks various crop infections automatically. This research paper shows how powerful these kinds of artificial intelligence could be when it comes to enhancing our farming methods. The authors introduce a CNN-based model that leverages clustering (specifically k-means) to improve the classification and early detection rate of rice diseases. As such, it produces efficient outcomes that are dependable though not very accurate and less time consuming. In this way, the size of the model is not underestimated, despite trading off some accuracy against it, hence potentially finding application in mobile and other real-time systems.K-means clustering is used to pick non-affected areas of rice leaf pictures for more accurate disease categorization. This technique divides pixels in an image into k clusters based on similarities, with each cluster characterized by a centroid. K-means clustering successfully separates diseased and non-affected sections of rice leaf images by iteratively assigning pixels to the nearest centroid and updating the centroids depending on the mean of the assigned pixels. This allows for accurate disease identification. The combination of k-means clustering with CNN improves the accuracy and efficiency of disease classification in rice crops.

The dataset used in the paper includes the "Rice Leaf Disease" dataset from the UCI Machine Learning Repository, the "Dhan-Shomadhan" dataset of Rice Leaf Disease Classification for Bangladeshi Local Rice, and the "Rice Leaf Disease Image Samples" dataset .

This study is separated into three stages: image acquisition, image pre-processing, and categorized images with a CNN model.

This paper used a Kaggle provided dataset containing three images of rice diseases, bacterial leaf blight(blue), brown spot(red), and leaf smut(green); there are 2700 training examples in the database out of which 59.6% belong to the training set while 13.9% belong to the testing set. Figure 2.5 shows the sequential steps of overall work in this paper.

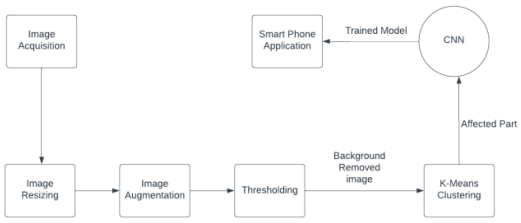


**Figure 2.5:** Sequential steps of overall work.

More information about the partitioned data is given in Figure 2.6[13].

| Class | Training Instances | Validation Instances | Testing Instances |
|---|---|---|---|
| Bacterial leaf blight | 900 | 400 | 210 |
| Brown spot | 900 | 400 | 210 |
| Leaf smut | 900 | 400 | 210 |

**Figure 2.6:** Number of images in different classes for training, testing and validation.

This paper has done image pre-processing techniques which includes background removal and extraction of the diseased portion using thresholding and k-means clustering. The RGB images are converted to HSV color space to isolate the S (saturation) channel for segmentation. Green, non-affected areas are replaced with white pixels to focus on the diseased sections.

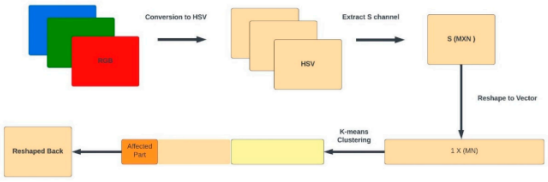


**Figure 2.7:** Fundamental block of using k-means clustering to select non-affected part.[13]

The CNN architecture incorporates several key layers. In addition, the model used for disease classification is designed to extract relevant features accurately classify rice leaf diseases. For feature extraction and disease classification, CNN model was

used in this paper by the authors.Incorporating layers for convolution, batch normalization, ReLU, pooling, dense networks, softmax, and classification, here CNN model was used. Convolutional layers apply convolutional filters to the input image, creating feature maps that capture spatial hierarchies and patterns essential for distinguishing between different diseases. Batch normalization technique normalizes the output of the convolution layers, improving the training speed and stability of the network. The ReLU activation function provides nonlinearity to the model, enabling it to learn complicated patterns by activating just the positive portions of the feature maps. Pooling layers, such as max-pooling, minimize the spatial dimensions of feature maps, reducing the computational cost while emphasizing the most relevant features. Dense fully connected layers connect every neuron in one layer to every neuron in the next layer, allowing the network to learn complex representations from the extracted features. The softmax layer[7] converts the output of the dense layers into probability distributions over the classes, enabling the classification of the input image into specific disease categories. This final layer of classification assigns a label to the input image based on the highest probability from the softmax layer[7]. The model's architecture and parameters are detailed in Figure 2.8. The model was run on Google Colaboratory with an Intel Xeon CPU, 12 GB RAM, and 103 GB disk space.

| Layer Type | Output Shape | No of Parameters |
|---|---|---|
| Conv3D | (None, 300, 300, 3, 4) | 112 |
| Batch Normalization | (None, 300, 300, 4) | 16 |
| Conv2D | (None, 300, 300, 32) | 1184 |
| Batch Normalization | (None, 300, 300, 32) | 128 |
| Conv2D | (None, 300, 300, 64) | 18,496 |
| Conv2D | (None, 300, 300, 64) | 36,928 |
| Batch Normalization | (None, 300, 300, 64) | 256 |
| Max Pooling 2D | (None, 300, 300, 64) | 0 |
| Conv2D | (None, 300, 300, 128) | 73,856 |
| Batch Normalization | (None, 300, 300, 128) | 512 |
| Max Pooling 2D | (None, 300, 300, 128) | 0 |
| Conv2D | (None, 300, 300, 256) | 295,168 |
| Batch Normalization | (None, 300, 300, 256) | 1024 |
| Conv2D | (None, 300, 300, 512) | 1,180,160 |
| Batch Normalization | (None, 300, 300, 512) | 2048 |
| Flatten | (None, 300, 300, 512) | 0 |
| Dense | (None, 300, 300, 512) | 6147 |
| Trainable Parameters: 3,974,875 | Non-Trainable Parameters: 3016 | Total: 3,977,891 |

**Figure 2.8:** No. of parameters employed in CNN network.[13]

The research presents a novel model for detecting and categorizing rice illnesses that combines image processing and Convolutional Neural Networks (CNN). The model has a high classification accuracy of 97.9% and a processing time of 103 ms, proving its efficacy in disease detection. The model's integration into desktop and mobile applications provides a feasible option for agricultural automation and remote sensing systems. This study makes a substantial contribution to increasing output from agriculture, improving food security, and extending the field of automated disease identification in crops. Furthermore, the model overtook existing approaches in terms

of accuracy along with processing speed, demonstrating its suitability for real-world agricultural applications.

## 2.6 Leveraging Pre-trained CNNs

In the study by Md. Shohanur Islam Sobuj et al.[34], the authors rigorously evaluate the impact of incorporating feature extraction methodologies within pre-trained CNNs for rice leaf disease classification. They focus on enhancing classification accuracy by employing advanced techniques such as Histogram of Oriented Gradients (HOG)[16], Local Binary Patterns (LBP), and Gradient-weighted Class Activation Mapping (Grad-CAM).

The dataset used in this paper is "Dhan-shomadhan: A dataset of rice leaf disease classification for Bangladeshi local rice," which comprises images of different rice diseases. This dataset provides a comprehensive collection of rice leaf images affected by various diseases, making it suitable for training and evaluating the proposed model.

The integration of advanced feature extraction techniques significantly improved the performance of various CNN architectures. The following techniques were employed:

HOG was used to capture the gradient structure of the images, which is particularly effective in highlighting disease-specific features. This technique notably boosted the accuracy of EfficientNet-B7 from 92% to an impressive 97%.

LBP was applied to capture the texture information of the rice leaf images. While the performance enhancements with LBP were more conservative compared to HOG, it still contributed to improved classification accuracy.

Grad-CAM was utilized to gain insights into the model's attention mechanisms. It showed that HOG integration led the model to focus more on disease-specific features, enhancing interpretability and confirming the effectiveness of the feature extraction techniques.

Visual representations validated the influence of HOG, exhibiting a clear increase in accuracy across epochs due to concentrated attention on diseased regions. These visual tools demonstrated how the model's focus on relevant areas led to better classification performance.

The study achieved a considerable accuracy of 97% using EfficientNet-B7 with HOG and Grad-CAM, marking a substantial improvement in optimizing pre-trained CNN-based rice disease identification methods. These findings underscore the importance

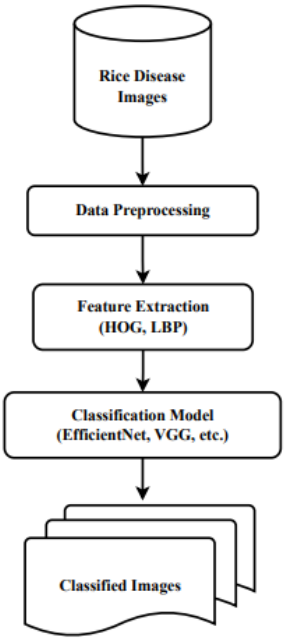


**Figure 2.9:** Proposed Workflow for Rice Disease Classification[34]

of combining advanced feature extraction techniques with state-of-the-art CNN architectures.

The findings advocate for the strategic integration of improved feature extraction approaches with innovative pre-trained CNN architectures. This method offers a promising path for significantly enhancing the accuracy and efficacy of image-based disease classification systems in agricultural applications.

In summary, leveraging pre-trained CNNs for efficient feature extraction in rice leaf disease classification represents a significant step forward in agricultural research. By incorporating advanced techniques such as HOG and Grad-CAM, researchers can achieve higher accuracy and efficiency in disease detection, contributing to increased agricultural productivity and food security. This study distinguishes itself by focusing on rice plant disease classification for Bangladeshi rice, utilizing sophisticated feature extraction techniques like HOG and LBP combined with pre-trained CNNs. The paper demonstrates how these strategies improve classification accuracy across various CNN architectures. Additionally, the integration of Grad-CAM enhances interpretability by highlighting critical regions in rice images for disease verification, providing valuable insights into the classification process.

# Chapter 3

# Proposed Methodology

In our proposed pipeline, we begin by inputting an image into an image enhancement model, specifically SRGAN. This model produces a high-quality image, which is then sent to a YOLO-based object detection model. The detected regions are cropped and sent to a transfer learning-based feature extractor. Finally, these extracted features are fed into a classifier network, which identifies the exact class of disease present in the input image.

## 3.1 Image Enhancement

### 3.1.1 SRGAN

Super-resolution Generative Adversarial Network (SRGAN) [38] is a deep learning-based approach for image super-resolution, which is the process of enhancing the resolution of an image. The methodology behind SRGAN involves the use of Generative Adversarial Networks (GANs) to generate high-resolution (HR) images from low-resolution (LR) inputs.

**Network Architecture**

- **Generator Network**

  The generator network is responsible for converting LR images into HR images. It typically consists of:

  - **Residual Blocks:** These blocks help in learning the identity mappings and improve the flow of information. Each residual block usually contains convolutional layers, batch normalization, and ReLU activation functions.

  - **Upsampling Layers:** These layers increase the spatial resolution of the image.
  - **Final Convolutional Layer:** This layer outputs the HR image.

- **Discriminator Network**

  The discriminator network distinguishes between real HR images and the generated HR images. It is usually a deep convolutional neural network (CNN) with several convolutional layers, batch normalization, and Leaky ReLU activations, followed by fully connected layers and a sigmoid activation function to output a probability score.

### Loss Functions

SRGAN employs a combination of different loss functions to train the networks:

- **Content Loss:** This loss measures the difference between the generated HR image and the original HR image. Typically, the VGG-based perceptual loss is used, which calculates the difference in high-level feature representations obtained from a pre-trained VGG network.
- **Adversarial Loss:** This loss comes from the discriminator and encourages the generator to produce images that are indistinguishable from real HR images. It is based on the binary cross-entropy between the discriminator's predictions and the actual labels.
- **Total Loss:** The total loss for the generator is a weighted sum of the content loss and the adversarial loss.

### Training Process

The training of SRGAN involves the following steps:

- **Pre-training the Generator:** Initially, the generator is pre-trained using content loss only. This helps the generator produce reasonable HR images before adversarial training.
- **Adversarial Training:** After pre-training, both the generator and the discriminator are trained simultaneously. The generator tries to produce HR images that can fool the discriminator, while the discriminator tries to correctly identify real and generated images.

### Evaluation Metrics

To evaluate the performance of SRGAN, several metrics are used:

- **Peak Signal-to-Noise Ratio (PSNR):** Measures the similarity between the generated HR image and the original HR image.
- **Structural Similarity Index (SSIM):** Measures the structural similarity between the generated HR image and the original HR image.
- **Visual Quality Assessment:** Subjective evaluation by humans to assess the perceptual quality of the generated images.

### Implementation Details

Key implementation details for SRGAN include:

- **Data Preparation:** Collecting and preparing datasets of LR-HR image pairs.
- **Hyperparameters:** Setting appropriate learning rates, batch sizes, and other hyperparameters for training.
- **Frameworks:** Using deep learning frameworks PyTorch to implement and train the networks.

SRGAN has been widely adopted and improved upon in various applications, providing significant advancements in image super-resolution tasks. enddocument

### Advantages of SRGAN over Other GANs

- **High-Resolution Image Generation:** SRGAN is specifically designed for the task of image super-resolution, which aims to reconstruct high-resolution images from low-resolution inputs. It is particularly effective in generating high-quality, detailed images.
- **Perceptual Loss:** Unlike traditional GANs, SRGAN uses a perceptual loss function that combines content loss and adversarial loss. The content loss is based on high-level feature representations from a pre-trained VGG network, leading to more visually pleasing results.
- **Adversarial Training:** SRGAN incorporates adversarial training that helps in generating more realistic and natural-looking textures in high-resolution images. This helps in overcoming the limitations of traditional loss functions that may lead to overly smooth images.

- **Detail Preservation:** The architecture of SRGAN includes residual blocks that help in preserving finer details and textures in the upsampled images. This results in high-quality images with less blurring and artifacts.
- **Generative Capabilities:** SRGAN leverages the power of GANs to generate high-resolution images that are not only accurate in terms of pixel values but also rich in perceptual quality, making them more appealing to human observers.
- **Versatility:** While SRGAN is tailored for super-resolution, its underlying principles can be adapted for other image-to-image translation tasks, making it a versatile model in the field of computer vision.
- **Community and Research Support:** SRGAN has gained significant attention and support from the research community, leading to continuous improvements and variations such as Enhanced SRGAN (ESRGAN) that further enhance its performance and robustness.

### 3.1.2 Lite SRGAN

Lite SRGAN[32] is a variant of SRGAN designed to reduce computational complexity and improve efficiency, making it suitable for deployment on resource-constrained devices such as mobile phones and embedded systems.

#### Optimizations in Lite SRGAN

Lite SRGAN introduces several optimizations to achieve a lightweight model:

- **Compact Network Design:** Lite SRGAN uses a more compact network architecture with fewer layers and parameters compared to the original SRGAN.
- **Efficient Residual Blocks:** Simplified residual blocks are used to reduce computational cost while maintaining performance.
- **Depthwise Separable Convolutions:** These convolutions are employed to decrease the number of parameters and operations.
- **Knowledge Distillation:** This technique is used to transfer knowledge from a larger teacher model to the smaller Lite SRGAN model.

#### Training Process for Lite SRGAN

The training process for Lite SRGAN is similar to that of SRGAN, with additional steps to ensure the model remains lightweight:

- **Pre-training:** Lite SRGAN undergoes pre-training using content loss to initialize the network.
- **Adversarial Training:** The model is then trained adversarially to refine the generated HR images.
- **Knowledge Distillation:** During training, knowledge distillation is used to transfer the learning from a larger pre-trained model to the Lite SRGAN model.

### Evaluation Metrics

To evaluate the performance of Lite SRGAN, several metrics are used:

- **Peak Signal-to-Noise Ratio (PSNR):** Measures the similarity between the generated HR image and the original HR image.
- **Structural Similarity Index (SSIM):** Measures the structural similarity between the generated HR image and the original HR image.
- **Visual Quality Assessment:** Subjective evaluation by humans to assess the perceptual quality of the generated images.
- **Computational Efficiency:** Metrics such as the number of parameters, FLOPs (Floating Point Operations), and inference time are used to evaluate the efficiency of Lite SRGAN.

### Implementation Details

Key implementation details for Lite SRGAN include:

- **Data Preparation:** Collecting and preparing datasets of LR-HR image pairs.
- **Hyperparameters:** Setting appropriate learning rates, batch sizes, and other hyperparameters for training.
- **Frameworks:** Using deep learning frameworks like TensorFlow or PyTorch to implement and train the networks.

Lite SRGAN, with its lightweight optimizations, provides an efficient solution for image super-resolution tasks on resource-constrained devices.

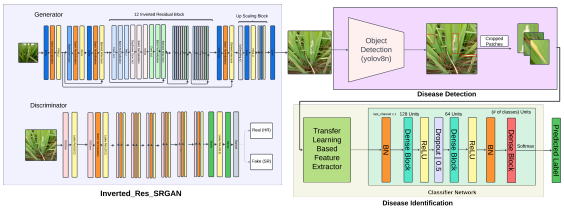


**Figure 3.1:** Proposed Workflow for Rice Disease Classification[34]

### 3.1.3 SRGAN with Inverted Residual Blocks and Depthwise Separable Convolutions SRGAN (Super-Resolution Generative Adversarial Network)

Aims to upscale low-resolution images to a higher resolution while preserving details. It consists of two sub-networks: a generator (G) and a discriminator (D). Generator (G): Uses inverted residual blocks, a variation of residual blocks, that might improve training efficiency by alleviating the vanishing gradient problem. Discriminator (D): Employs depthwise separable convolutions, which factorize traditional convolutions into depthwise (applying filters to each channel) and pointwise (combining filtered channels) convolutions, potentially reducing computational cost.

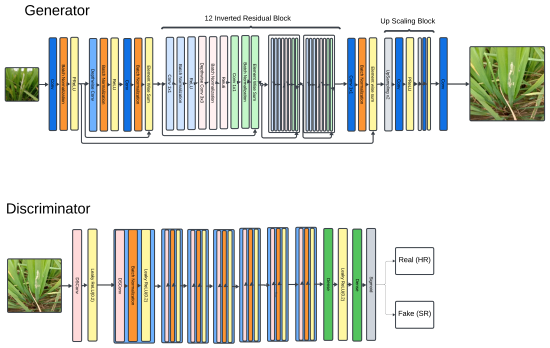


**Figure 3.2:** Proposed Workflow for Rice Disease Classification[34]

**Generator Network**

**Advantages of Inverted Residual Blocks over Residual Blocks in SRGAN Generator**

## Residual Block

**Definition:** A Residual Block is defined as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}, \tag{3.1}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the input and output, respectively, and $\mathcal{F}(\mathbf{x}, \{W_i\})$ represents the residual mapping to be learned.

**Advantages:**

- **Ease of Optimization:** Residual blocks allow for easier optimization by providing shortcuts for gradient flow, thus mitigating the vanishing gradient problem.
- **Identity Mapping:** The identity mapping helps preserve information from the input, leading to better performance in deep networks.

## Inverted Residual Block

**Definition:** An Inverted Residual Block can be mathematically expressed as:

$$\mathbf{y} = \mathbf{x} + \mathcal{F}(\mathbf{x}, \{W_i\}), \tag{3.2}$$

where the transformation $\mathcal{F}$ consists of a sequence of layers that expand the number of channels (using a pointwise convolution), apply depthwise convolution, and then project back to a lower-dimensional space.

**Components:**

- **Expansion:** The input tensor is first expanded by a factor $t$ using a $1 \times 1$ convolution.
- **Depthwise Convolution:** A depthwise separable convolution is applied, which significantly reduces computational complexity.
- **Projection:** The tensor is projected back to the original dimension using another $1 \times 1$ convolution.

**Advantages:**

- **Efficiency:** Inverted residual blocks reduce the number of parameters and computational cost by separating the spatial and channel-wise computations.
- **Non-linearity:** The expansion phase introduces non-linearity in a higher–dimensional space, which allows the network to capture more complex features.
- **Feature Utilization:** By using depthwise convolutions, the block can better utilize spatial features, leading to more efficient representation learning.

## Logical and Mathematical Advantages

- **Reduced Computational Cost:** The depthwise separable convolutions in the inverted residual block decrease the number of operations from $O(n^2 \cdot k^2 \cdot c^2)$ to $O(n^2 \cdot k^2 \cdot c)$, where $n$ is the spatial dimension, $k$ is the kernel size, and $c$ is the number of channels.
- **Better Gradient Flow:** Similar to residual blocks, inverted residual blocks maintain good gradient flow, but the expanded intermediate representation allows for more expressive features to be learned.
- **Parameter Efficiency:** Inverted residual blocks use fewer parameters to achieve similar or better performance than traditional residual blocks, making them suitable for resource-constrained environments.
- **Improved Feature Representation:** The expansion phase allows the network to learn more diverse features by operating in a higher-dimensional space temporarily.

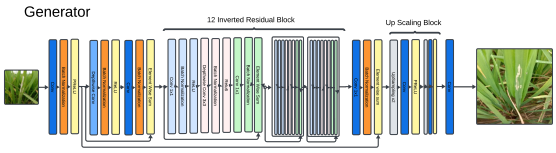


**Figure 3.3:** Proposed Workflow for Rice Disease Classification[34]

- **Input Layer:**
  - Input: Low-quality image.
- **Initial Convolutional Block:**
  - Convolutional layer with 64 filters, kernel size 9×9, stride 1, padding same.
  - Batch Normalization layer.
  - Parametric ReLU (PReLU) activation layer.
- **Inverted Residual Blocks:**
  - A series of 12 inverted residual blocks, each consisting of:
    - Depthwise Convolutional layer, kernel size $3 \times 3$, stride 1, padding same.
    - Batch Normalization layer.

        * ReLU activation layer.
        * Convolutional layer with expansion and projection, kernel size $1 \times 1$, stride 1, padding same.
        * Batch Normalization layer.
        * Element-wise summation of the output with the input of the ReLU layer.

- **Convolutional Layer:**
  - Convolutional layer with 64 filters, kernel size $3 \times 3$, stride 1, padding same.
  - Batch Normalization layer.
- **Upscaling Blocks:**
  - Two sub-pixel convolutional layers, each consisting of:
    * Convolutional layer with 256 filters, kernel size $3 \times 3$, stride 1, padding same.
    * PixelShuffle layer for upscaling (factor 2).
    * Parametric ReLU (PReLU) activation layer.
- **Output Convolutional Layer:**
  - Convolutional layer with 3 filters (RGB), kernel size $9 \times 9$, stride 1, padding same.
  - Tanh activation layer.

**Discriminator Network**

**Advantages of Depthwise Convolutional Blocks over Traditional Convolutional Blocks in SRGAN Discriminator**

## Traditional Convolutional Block

**Definition:** A traditional convolutional block performs a standard convolution operation on the input tensor. Mathematically, it is represented as:

$$\mathbf{y} = \mathcal{C}(\mathbf{x}, W), \tag{3.3}$$

where $\mathbf{x}$ is the input tensor, $W$ is the convolutional filter, and $\mathcal{C}$ represents the convolution operation.

**Advantages:**

- **Feature Extraction:** Effective in extracting both spatial and channel-wise features simultaneously.
- **Versatility:** Widely used and well-understood, with strong performance in various image processing tasks.

## Depthwise Convolutional Block

**Definition:** A depthwise convolutional block separates the convolution operation into two steps: depthwise convolution and pointwise convolution. Mathematically, it can be expressed as:

$$\mathbf{y} = \mathcal{P}(\mathcal{D}(\mathbf{x}, W_d), W_p), \tag{3.4}$$

where $\mathcal{D}$ represents the depthwise convolution with filter $W_d$, and $\mathcal{P}$ represents the pointwise convolution with filter $W_p$.

**Components:**

- **Depthwise Convolution:** Applies a single convolutional filter per input channel.
- **Pointwise Convolution:** Applies a $1 \times 1$ convolution to combine the outputs of the depthwise convolution.

**Advantages:**

- **Efficiency:** Reduces the number of parameters and computational cost by performing separate spatial and channel-wise convolutions.
- **Feature Utilization:** Better utilization of spatial features through depthwise convolution, which captures spatial dependencies within each channel.
- **Non-linearity:** Introducing non-linearity after the depthwise convolution allows for more complex feature learning.

## Logical and Mathematical Advantages

- **Reduced Computational Cost:** The computational cost of a traditional convolutional block is $O(n^2 \cdot k^2 \cdot c^2)$, where $n$ is the spatial dimension, $k$ is the kernel size, and $c$ is the number of channels. For a depthwise convolutional block, the cost is reduced to:

$$O(n^2 \cdot k^2 \cdot c + n^2 \cdot c \cdot c) = O(n^2 \cdot k^2 \cdot c + n^2 \cdot c^2). \tag{3.5}$$

This separation leads to significantly fewer operations, especially for large $c$ and $k$.

- **Parameter Efficiency:** Traditional convolutional layers require $k^2 \cdot c^2$ parameters, whereas depthwise separable convolutions require $k^2 \cdot c + c^2$ parameters. This reduction in parameters leads to more efficient models.
- **Improved Feature Representation:** Depthwise convolutions focus on capturing spatial information within each channel independently, while pointwise convolutions capture inter-channel dependencies. This separation allows for a richer and more detailed feature representation.
- **Enhanced Non-linearity:** By introducing non-linearity between depthwise and pointwise convolutions, the model can learn more complex and diverse features, improving the discriminator's ability to distinguish between real and generated images.
- **Regularization Effect:** The separation of convolutions acts as a form of regularization, reducing overfitting by limiting the co-adaptation of filters, which can improve generalization performance.

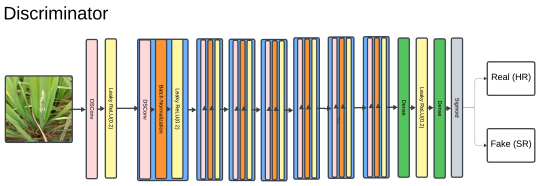


**Figure 3.4:** Proposed Workflow for Rice Disease Classification[34]

- **Input Layer:**
  - Input: High-resolution or generated image.
- **Convolutional Blocks:**
  - 8 blocks, each consisting of:
    - Convolutional layer with increasing filter sizes, kernel size $3\times3$, stride 1 or 2 (for downsampling), padding same.
    - Batch Normalization layer.
    - Leaky ReLU activation layer ($\alpha = 0.2$).
- **Fully Connected Layers:**
  - Dense layer with 1024 units.

- Leaky ReLU activation layer ($\alpha = 0.2$).
- Dense layer with 1 unit.
- Sigmoid activation layer for binary classification (real or fake).

This proposed architecture leverages inverted residual blocks within the SRGAN framework to enhance image quality, incorporating depthwise separable convolutions and efficient upscaling techniques. The discriminator network robustly distinguishes between real and generated images through multiple convolutional and fully connected layers.

In Figure 4.1, the qualitative and quantitative comparison has been shown between the basic SRGAN model and the Lite SRGAN model. It can be said that the comparison shows similar effects by both models. The PSNR and SSIM values for each show that both give a fairly high-quality image. The experiment was done using the Dhan-Shamadhan dataset and trained up to 300 epochs.

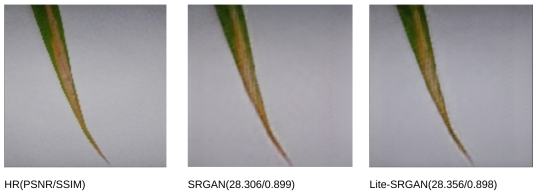


**Figure 3.5:** Image Enhancement Performance with SRGAN and Lite SRGAN

## 3.2 Object Detection

We have used different versions of YOLO models for object detection. Among them, we got the best results from YOLOv8n architecture. With took help from some researchers and agriculturists for the annotation of our dataset images.

### 3.2.1 Data Annotation

With the help of BRRI and Researchers and Agriculturists from Sher–e–Bangla Agricultural University we have annotated our data for object detection model training.

### 3.2.2 YOLOv8x

YOLOv8x [11] is a state-of-the-art object detection model belonging to the You Only Look Once (YOLO) family, which is known for its balance of speed and accuracy. YOLOv8x builds upon the advancements of previous YOLO versions, introducing several enhancements in architecture and training techniques to achieve superior performance in detecting objects in images.

#### Architecture

The architecture of YOLOv8x incorporates several key innovations:

- **Backbone Network**:
  - Utilizes a CSP-Darknet53 backbone, which is a variation of Darknet with Cross Stage Partial connections. This design improves gradient flow and reduces computation.
  - Employs deeper and wider layers compared to its predecessors to capture more complex features.
- **Neck Network**:
  - Implements a Path Aggregation Network (PANet) to enhance feature pyramid networks. PANet improves information flow from the backbone to the head, facilitating better multi-scale feature fusion.
  - Uses SPP (Spatial Pyramid Pooling) to handle objects at different scales by pooling features from multiple receptive fields.
- **Head Network**:
  - Features an anchor-free design, which simplifies the network and reduces computational overhead.
  - Employs decoupled heads for classification and regression tasks, allowing the model to specialize and improve accuracy.

#### Method

YOLOv8x employs several advanced training techniques to boost performance:

- **Data Augmentation**:
  - Utilizes techniques such as mosaic augmentation, which combines four training images into one, providing more diverse training samples.

- Applies mixup augmentation, blending two images together to improve the model's robustness to occlusion and varying lighting conditions.

- **Loss Function**:
  - Adopts a combination of classification, localization, and objectness loss to optimize the model effectively.
  - Incorporates CIoU (Complete Intersection over Union) loss for bounding box regression, which considers the distance between boxes, aspect ratio, and overlap.

- **Hyperparameter Optimization**:
  - Implements automated hyperparameter tuning to find the optimal set of parameters for training.
  - Uses advanced optimization algorithms like AdamW to improve convergence and generalization.

In summary, YOLOv8x represents a significant advancement in object detection technology, combining innovative architectural design with cutting-edge training techniques to achieve high accuracy and efficiency.

### 3.2.3 YOLOv8n

#### Architecture

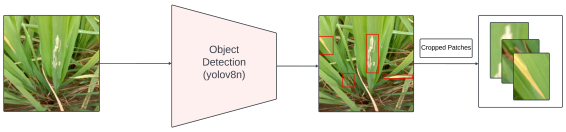


**Figure 3.6:** Disease localization using Light weight Object Detection model

YOLOv8n [15] (YOLO version 8, nano) is designed to be a lightweight and efficient object detection model. Its architecture follows the typical YOLO structure but is optimized for speed and low computational resources. The architecture can be broken down into several key components: the Backbone, the Neck, and the Head.

#### 1. Backbone

The backbone is responsible for feature extraction from the input image. In YOLOv8n, a series of convolutional layers are used, often incorporating novel elements like CSP (Cross Stage Partial) connections to improve gradient flow and reduce computational complexity.

**Key Components:**

- **Convolutional Layers:** These layers apply convolutional filters to the input image, extracting basic features such as edges and textures.
- **CSP Connections:** CSP connections split the feature map into two parts, applying transformations to one part while the other part remains unchanged, then merging them. This helps in reducing the number of parameters and improving gradient flow.
- **Residual Blocks:** Residual connections help in training deeper networks by providing shortcuts for gradient flow.

**Mathematical Representation:**

$$\mathbf{F}_i = \sigma(\mathbf{W}_i * \mathbf{X} + \mathbf{b}_i), \tag{3.6}$$

where $\mathbf{F}_i$ is the feature map at layer $i$, $\mathbf{W}_i$ and $\mathbf{b}_i$ are the weights and biases of the convolutional layer, $\mathbf{X}$ is the input, $*$ denotes convolution, and $\sigma$ is the activation function.

**2. Neck**

The neck of YOLOv8n aggregates features from different levels of the backbone to better detect objects at various scales. This often involves PANet (Path Aggregation Network) or FPN (Feature Pyramid Network) structures.

**Key Components:**

- **PANet/FPN:** These networks fuse features from different layers to enhance the representation of objects at multiple scales.
- **Upsampling and Concatenation:** Features from deeper layers are upsampled and concatenated with features from earlier layers to combine spatial and semantic information.

**Mathematical Representation:**

$$\mathbf{F}_{\text{agg}} = \text{Concat}(\text{Upsample}(\mathbf{F}_l), \mathbf{F}_s), \tag{3.7}$$

where $\mathbf{F}_l$ are the features from a deeper layer, $\mathbf{F}_s$ are the features from a shallower layer, and Concat denotes concatenation.

**3. Head**

The head of YOLOv8n is responsible for predicting bounding boxes, objectness scores,

and class probabilities. This typically involves a series of convolutional layers that reduce the feature maps to the final detection outputs.

**Key Components:**

- **Detection Layers:** These layers predict bounding box coordinates, objectness score, and class probabilities for each anchor box.
- **Anchor Boxes:** Predefined boxes of different shapes and sizes used to predict the final bounding boxes.

**Mathematical Representation:**

$$\mathbf{P} = \sigma(\mathbf{W}_{\text{cls}} * \mathbf{F}_{\text{neck}} + \mathbf{b}_{\text{cls}}), \tag{3.8}$$

where $\mathbf{P}$ represents the predictions, $\mathbf{W}_{\text{cls}}$ and $\mathbf{b}_{\text{cls}}$ are the weights and biases for the final convolutional layer, and $\mathbf{F}_{\text{neck}}$ is the aggregated feature map from the neck.

**Logical Flow**

1. **Input Image:** An image of size $H \times W \times 3$ is fed into the model.
2. **Backbone:** The backbone processes the image through convolutional layers and CSP connections, extracting feature maps.
3. **Neck:** The neck aggregates and enhances these feature maps using PANet/FPN structures.
4. **Head:** The head processes the aggregated features to predict bounding boxes, objectness scores, and class probabilities.
5. **Output:** The final output consists of bounding boxes, objectness scores, and class probabilities for detected objects.

### Improving Disease Identification Using YOLOv8n for Localization and Cropping

Using YOLOv8n to localize disease-affected regions and then cropping these patches for further identification can significantly improve the accuracy of the identification process. This approach leverages the strength of YOLOv8n in object detection to focus the identification model on relevant areas, ensuring that it processes the most informative parts of the image.

**Disease Part Localization using YOLOv8n (You Only Look Once version 8 nano)**

A real-time object detection model known for its speed and accuracy. You'll likely use a pre-trained YOLOv8n model to detect the diseased area in the SRGAN-enhanced image. YOLOv8n's ability to achieve real-time object detection on resource-constrained platforms stems from several key design choices that promote its lightness and computational efficiency:

- **Smaller Model Size:** Compared to larger YOLO versions (e.g. YOLOv8x), YOLOv8n utilizes fewer filters and channels within its convolutional layers. This directly reduces the number of parameters the model needs to learn, making it more compact and requiring less memory during inference.
- **Focus on Efficient Operations:**
  - **Depthwise Separable Convolutions:** These convolutions decompose the traditional convolution operation into two separate steps: depthwise (applying filters to each channel independently) and pointwise (combining filtered channels). This factorization often requires fewer computations compared to standard convolutions.
  - **Bottleneck Layers:** These are a type of residual block that incorporate a lower-dimensional hidden layer within the block structure. This design choice reduces the number of parameters and computations needed while maintaining the model's representational power for learning complex features.
- **Simplified Architecture:** YOLOv8n employs a streamlined architecture compared to more complex object detection models. It utilizes fewer building blocks and avoids redundant operations within the network, leading to faster execution times during inference.
- **Quantization (Optional):** While not always used by default, YOLOv8 can be quantized to lower precision formats (e.g., from float32 to int8). Quantization reduces the memory footprint of the model and enables faster inference on devices with limited resources, further enhancing its computational efficiency.

  In summary, YOLOv8n achieves its lightweight and efficient nature through a combination of the following: Smaller model size, Focus on efficient operations (depthwise separable convolutions, bottleneck layers), Simplified architecture, Potential for quantization

**Process Description**

- **Localization with YOLOv8n:** YOLOv8n is used to detect and localize disease-affected regions within an input image. The model outputs bounding boxes around these regions.
- **Cropping Disease Patches:** The bounding boxes provided by YOLOv8n are used to crop the disease-affected regions from the original image. These cropped patches are then used for further analysis.
- **Identification Model:** The cropped patches are passed to a specialized identification model that focuses on classifying the type of disease. This model can be a convolutional neural network (CNN) or any other suitable architecture for image classification.

### 3.2.4 Advantages of the Approach

**1. Focused Feature Extraction**

**Logic:** By localizing and cropping the disease-affected regions, the identification model can focus on the most relevant parts of the image. This reduces the noise and irrelevant background information that the model has to process.

**Mathematical Representation:**

$$\mathbf{I}_{\text{crop}} = \mathbf{I}(\mathbf{B}), \tag{3.9}$$

where $\mathbf{I}$ is the original image, $\mathbf{B}$ are the bounding boxes provided by YOLOv8n, and $\mathbf{I}_{\text{crop}}$ are the cropped patches.

**2. Improved Accuracy**

**Logic:** Focusing on disease-affected regions ensures that the identification model is not distracted by healthy tissue or irrelevant parts of the image. This increases the signal-to-noise ratio, allowing the model to learn and recognize disease features more effectively.

**Mathematical Representation:** Let $\mathbf{X}_{\text{full}}$ be the input features from the entire image and $\mathbf{X}_{\text{crop}}$ be the input features from the cropped patches. The accuracy improvement can be expressed as:

$$\text{Accuracy}(\mathbf{X}_{\text{crop}}) > \text{Accuracy}(\mathbf{X}_{\text{full}}). \tag{3.10}$$

**3. Computational Efficiency**

**Logic:** Processing smaller, localized patches is computationally more efficient than processing the entire image. This efficiency can lead to faster training and inference times for the identification model.

**Mathematical Representation:** Let $C(\mathbf{I})$ be the computational cost for processing the entire image and $C(\mathbf{I}_{\text{crop}})$ be the cost for processing the cropped patches. The efficiency can be expressed as:

$$C(\mathbf{I}_{\text{crop}}) < C(\mathbf{I}). \tag{3.11}$$

**4. Enhanced Model Interpretability**

**Logic:** By focusing on specific disease-affected regions, the identification model's predictions become more interpretable. Clinicians can better understand the model's decisions and verify that it is focusing on the correct areas.

**Mathematical Representation:** The interpretability can be measured by the overlap between the model's attention areas and the actual disease regions:

$$\text{IOU}(\mathbf{B}, \mathbf{G}) \approx 1, \tag{3.12}$$

where IOU is the Intersection over Union between the predicted bounding boxes $\mathbf{B}$ and the ground truth boxes $\mathbf{G}$.

### Conclusion

Using YOLOv8n to localize and crop disease-affected regions before passing them to an identification model enhances the accuracy and efficiency of disease identification. This approach ensures that the identification model focuses on the most relevant features, leading to better performance and more interpretable results.

## 3.3 Disease Identification

At first, we ran different Neural Network architectures like- Basic CNN, MobileNet, VGG19, and EffiecientNet on different datasets using a learning rate=0.001, batch size=64, epochs=60, and optimizer=Adam.

A **Basic Convolutional Neural Network (CNN)**[20] is a type of deep neural network primarily used for analyzing visual data. It consists of several layers including convolutional layers, pooling layers, and fully connected layers. Convolutional layers

apply filters to the input image to create feature maps, pooling layers reduce the dimensionality of these feature maps, and fully connected layers interpret these features to make predictions. Basic CNNs are foundational models used in various computer vision tasks such as image classification and object detection.

**MobileNet**[9], [22] is a class of efficient models designed for mobile and embedded vision applications. It uses depthwise separable convolutions to reduce the number of parameters and computational costs, making it suitable for devices with limited resources. MobileNet achieves a good balance between latency and accuracy, making it ideal for applications requiring real-time processing.

**VGG19**[39] is a deep convolutional neural network that is 19 layers deep, developed by the Visual Geometry Group (VGG) at Oxford. It is characterized by its use of very small (3x3) convolution filters, which are stacked to increase the depth of the network. VGG19 is known for its simplicity and effectiveness in image classification tasks. Despite its relatively large size and computational cost, it has been widely used due to its excellent performance.

**EfficientNet**[21] is a family of convolutional neural networks developed by Google AI, which scales the network dimensions (depth, width, and resolution) using a compound scaling method. EfficientNet models achieve state-of-the-art accuracy while being more computationally efficient than previous models. This efficiency is achieved through a carefully balanced scaling method that uniformly scales all dimensions of the network, resulting in better performance with fewer parameters and lower computational cost.

These experiments didn't give satisfactory results. So, later additionally, we also ran MixNet, XceptionNet, and InceptionNet on these datasets using learning rate=0.001, batch size=64, epochs=100, and optimizer=Adam.

**MixNet**[24] is a family of neural networks designed for efficient image classification, developed by Google AI. MixNet introduces the concept of "mixed depthwise convolution," where multiple filter sizes are mixed in a single convolutional layer. This allows the network to capture diverse feature patterns efficiently. MixNet models are designed to improve the trade-off between accuracy and computational cost, making them suitable for mobile and edge devices.

**XceptionNet**[5] (Extreme Inception) is a deep convolutional neural network that builds upon the Inception architecture but replaces the standard Inception modules with depthwise separable convolutions. This design simplifies the model and improves performance by enabling more efficient feature extraction. XceptionNet achieves

high accuracy on image classification tasks while being computationally efficient, making it a popular choice for various computer vision applications.

**InceptionNet**[36], also known as GoogleNet, is a deep convolutional neural network architecture introduced by Google. It uses inception modules, which apply multiple convolution filters of different sizes to the same input, allowing the network to capture features at various scales simultaneously. InceptionNet is known for its ability to achieve high accuracy with relatively low computational cost. The architecture has evolved through several versions, including Inception-v1, Inception-v2, Inception-v3, and Inception-v4, each introducing improvements and optimizations.

Though these experiments gave us better results, this still was not as per our expectations. Finally, we decided to try the ViT models and the results showed remarkable progress.

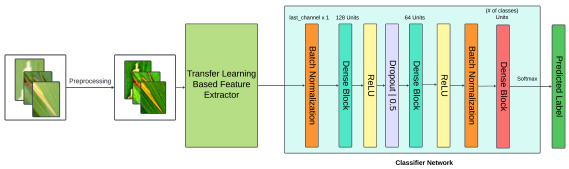


**Figure 3.7:** Disease Identification with Transfer Learning-based model with classifier Network

### 3.3.1 Advantages of Vision Transformers (ViT)

#### 1. Logical Advantages of ViT

- **Global Context:** ViTs capture global context efficiently through self-attention mechanisms. Unlike Convolutional Neural Networks (CNNs), which have a limited receptive field and rely on local convolutions to capture features, ViTs can directly model long-range dependencies in the image.
- **Scalability:** ViTs scale well with increased data and model size. They have been shown to outperform CNNs when trained on large datasets, leveraging the power of transformers originally developed for Natural Language Processing (NLP).
- **Flexibility:** ViTs can process image patches of arbitrary sizes, making them more flexible in terms of input image dimensions and resolutions. This adaptability can lead to better generalization across different tasks and datasets.
- **Parameter Efficiency:** ViTs can achieve superior performance to CNNs with

fewer parameters, especially when pre-trained on large-scale datasets & fine-tuned on specific tasks.

**2. Mathematical Advantages of ViT**

**Self-Attention Mechanism:** The core of ViT is the self-attention mechanism, which can be mathematically represented as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{3.13}$$

where $Q$ (queries), $K$ (keys), and $V$ (values) are linear transformations of the input, and $d_k$ is the dimension of the keys. This allows the model to weigh the importance of different patches based on their content, capturing global dependencies.

**Patch Embedding:** ViT divides an image into patches and embeds them into a lower-dimensional space:

$$\mathbf{E}_i = \text{Linear}(\text{Flatten}(\mathbf{P}_i)), \tag{3.14}$$

where $\mathbf{P}_i$ is the $i$-th image patch. These embeddings are then processed by the transformer layers.

**Positional Encoding:** To retain spatial information, ViTs add positional encodings to the patch embeddings:

$$\mathbf{Z}_0 = [\mathbf{E}_1 + \mathbf{p}_1, \mathbf{E}_2 + \mathbf{p}_2, \ldots, \mathbf{E}_N + \mathbf{p}_N], \tag{3.15}$$

where $\mathbf{p}_i$ are the positional encodings for each patch.

### 3.3.2 Advantages of Adding a Classifier Network with Dense Layers

**1. Logical Advantages of the Classifier Network**

- **Feature Transformation:** Adding dense layers allows the network to transform and refine the features extracted by ViT, improving the discrimination power of the model.
- **Non-Linearity:** The ReLU activation introduces non-linearity, enabling the model to learn more complex decision boundaries.
- **Regularization:** Batch normalization and dropout help in regularizing the net-

work, preventing overfitting and improving generalization.

- **Hierarchical Representation:** Multiple dense layers can capture hierarchical representations, making the network more robust in distinguishing between different classes.

**2. Detailed Structure and Mathematical Representation**

The classifier network can be structured as follows:

- **Batch Normalization:** Normalizes the input to have zero mean and unit variance.
- **Dense Layer (128 units) + ReLU:**

$$\mathbf{h}_1 = \text{ReLU}(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1), \tag{3.16}$$

  where $\mathbf{W}_1$ and $\mathbf{b}_1$ are the weights and biases of the first dense layer, and $\mathbf{x}$ is the input.

- **Dropout:**

$$\mathbf{h}_1 = \text{Dropout}(\mathbf{h}_1), \tag{3.17}$$

  where a certain percentage of the activations in $\mathbf{h}_1$ are randomly set to zero during training.

- **Dense Layer (64 units) + ReLU:**

$$\mathbf{h}_2 = \text{ReLU}(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2), \tag{3.18}$$

  where $\mathbf{W}_2$ and $\mathbf{b}_2$ are the weights and biases of the second dense layer.

- **Dense Layer (Output Units = # of Classes):**

$$\mathbf{y} = \text{Softmax}(\mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3), \tag{3.19}$$

  where $\mathbf{W}_3$ and $\mathbf{b}_3$ are the weights and biases of the final dense layer, and the softmax function ensures the output is a probability distribution over the classes.

**3. Advantages of the Structure**

- **Improved Feature Representation:** The dense layers refine the features, enhancing the ability of the network to distinguish between classes.

- **Regularization and Stability:** Batch normalization stabilizes the training process, and dropout helps prevent overfitting.
- **Non-Linearity:** ReLU activations introduce non-linearities, allowing the model to learn complex patterns.
- **Class Prediction:** The final dense layer with softmax activation produces class probabilities, facilitating accurate classification.

# Chapter 4

# Results and Discussion

## 4.1 Dataset

### 4.1.1 Paddy Doctor: Paddy Disease Classification

**Training dataset:** 10,407 (75%) labeled paddy leaf images across ten classes (nine diseases and normal leaf).
**Test dataset:** 3,469 (25%) paddy leaf images into one of the nine diseases or normal leaf. [28]

#### Diverse Classes

The dataset encompasses 10 distinct categories, including various paddy diseases and healthy leaves, facilitating extensive research on disease identification.

#### Challenges and Opportunities

- **Imbalanced Classes** Like many real-world datasets, the Paddy Doctor dataset has imbalanced classes. Some diseases are more common than others, creating challenges for model training.
- **Leaf Variability** Leaves can vary due to factors like lighting conditions, growth stages, and their position. These variations must be considered when developing robust algorithms.
- **Generalization** It is crucial to develop models that generalize well across different paddy varieties, environmental conditions, and geographical locations.

While the Paddy Doctor dataset is a valuable resource, continuous research and collaboration are crucial to further the field of plant disease detection.

### 4.1.2 Dhan-Shomadhan: A Dataset of Rice Leaf Disease Classification for Bangladeshi Local Rice

- 5 different harmful diseases of rice leaf called Brown Spot, Leaf Scaled, Rice Blast, Rice Tungro, and Sheath Blight.
- Dataset contains 1106 images. [8]

#### Dataset Overview

The dataset contains 1106 images representing five harmful rice leaf diseases commonly found in Bangladesh. These diseases are:

- **Brown Spot**
- **Leaf Scald**
- **Rice Blast**
- **Rice Tungro**
- **Sheath Blight**

The images are captured in two different background variations:

- **Field background**
- **White background**

#### Applications

Researchers and practitioners can utilize this dataset for:

- **Rice leaf disease classification**: Developing models to automatically identify and classify different rice diseases.
- **Disease detection using Computer Vision**: Leveraging pattern recognition techniques to detect and diagnose rice leaf diseases.

In summary, this dataset contributes significantly to advancing research in rice disease detection, benefiting farmers, and promoting sustainable agriculture practices.

### 4.1.3 BRRI's Online Available Dataset

- total 1426 images of rice diseases and pests. The number of images in each class is shown in Table 4.1. [17]

**Table 4.1:** Image Collection of Different Classes (BRRI)

| Class Name | No. of Images Collected |
|---|---|
| false Smut | 93 |
| Brown Plant Hopper | 71 |
| Bacterial Leaf Blight | 138 |
| Neck Blast | 286 |
| Stemborer | 201 |
| Hispa | 73 |
| Sheath Blight/Shath Rot | 219 |
| Brown Spot | 111 |
| Others | 234 |

#### Data Collection

For making this dataset, field data collection was conducted under the supervision of experts from the Bangladesh Rice Research Institute (BRRI). The dataset aims to capture real-world variations in rice grain health.

#### Use Cases and Applications

Researchers and practitioners can leverage this dataset for:

- **Disease Classification**: Developing machine learning models to automatically identify and classify rice grain diseases.
- **Computer Vision Applications**: Using pattern recognition techniques for rice grain health assessment.
- **Precision Agriculture**: Supporting farmers in making informed decisions about disease management.

## 4.2 Our Experiments

We have done experiments using different models and datasets for both image enhancement and image classification. The details of our experiments are discussed in the following subsections:

### 4.2.1 Image Enhancement

In Figure 4.1 the qualitative and quantitative comparison has been shown between the basic SRGAN model & the Lite SRGAN model. It can be said that the comparison shows similar effects by both the models. The PSNR and SSIM values for each show that both give a fairly high-quality image. The experiment was done using the Dhan-Shamadhan dataset and trained upto 300 epochs.

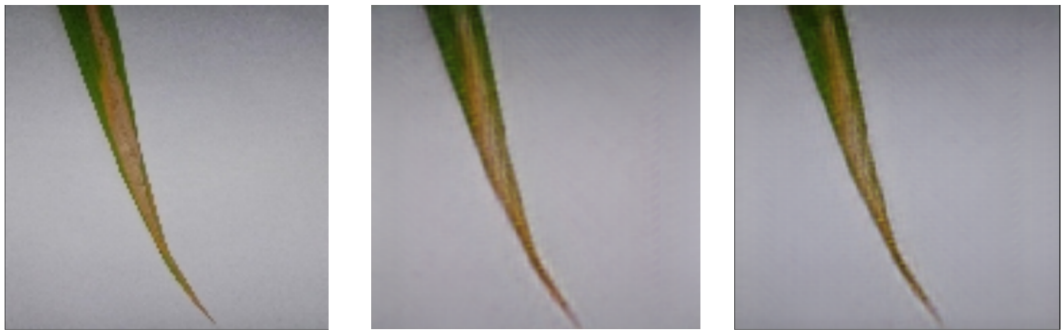


**Figure 4.1:** Image Enhancement Performance with SRGAN and Lite SRGAN

**Table 4.2:** SRGAN Model Analysis

| Dhan Shamadhan Dataset [8] No. of Classes : 10 Total images: 13,876 | | | | | |
|---|---|---|---|---|---|
| Model | Size (MB) | Inference time for a single image (ms) | Maximum GPU memory allocated (MB) | Params (M) | FLOPs (G) |
| SRGAN | 12.02 | 62.79 | 2201.90 | 19.49 | 27.96 |
| Our | 4.91 | 28.46 | 163.61 | 1.51 | 21.47 |

**Explanation**

The table labeled "Table 4.2: SRGAN Model Analysis" provides a comparative analysis between two models: the SRGAN model and a proposed model (labeled as "Our")

based on the Dhan Shamadhan Dataset. The comparison is made on several parameters, which are listed below:

- **Model**: This column lists the names of the models being compared (SRGAN and Our).
- **Size (MB)**: This column indicates the size of each model in megabytes (MB).
  - SRGAN: 12.02 MB
  - Our: 4.91 MB

  The proposed model ("Our") is significantly smaller in size compared to the SRGAN model, indicating that it is more compact and likely to be more efficient in terms of storage.
- **Inference time for a single image (ms)**: This column shows the time (in milliseconds) taken by each model to process a single image.
  - SRGAN: 62.79 ms
  - Our: 28.46 ms

  The proposed model processes a single image much faster than the SRGAN model, indicating better real-time performance and efficiency.
- **Maximum GPU memory allocated (MB)**: This column provides the maximum amount of GPU memory allocated by each model in megabytes (MB).
  - SRGAN: 2201.90 MB
  - Our: 163.61 MB

  The proposed model uses significantly less GPU memory compared to the SRGAN model, which means it is more memory-efficient and can run on devices with limited GPU resources.
- **Params (M)**: This column lists the number of parameters in each model, measured in millions (M).
  - SRGAN: 19.49 M
  - Our: 1.51 M

  The proposed model has far fewer parameters than the SRGAN model, suggesting that it is less complex and potentially easier to train.

- **FLOPs (G)**: This column shows the number of floating-point operations per second (FLOPs) in billions (G) for each model.
  - SRGAN: 27.96 G
  - Our: 21.47 G

The proposed model has fewer floating-point operations per second, indicating it requires less computational power compared to the SRGAN model.

**Summary**

The comparison table highlights the advantages of the proposed model over the SRGAN model in terms of size, inference time, GPU memory allocation, number of parameters, and computational efficiency. The proposed model ("Our") is more compact, faster, more memory-efficient, and less computationally intensive, making it a potentially better choice for practical applications, especially in environments with limited computational resources.

### 4.2.2 Object Detection

**Table 4.3:** Object Detection Results

| Model | Model Size(MB) | mAP50 | mAP50-95 | Params(M) | FLOPs(G) |
|---|---|---|---|---|---|
| **yolov8n** | **6.23** | 0.5905 | 0.3348 | **0.91** | **1.08** |
| **yolov8x** | 130 | **0.6990** | **0.4254** | 99.1 | 281.9 |

**Model Comparison**

The table 4.3 provides a comparison of two object detection models, YOLOv8n and YOLOv8x, based on several performance metrics. Here's a breakdown of each column and what they represent:

- **Model**: The name of the model being compared.
- **Model Size**: The size of the model in megabytes (MB).
- **mAP50**: Mean Average Precision at IoU (Intersection over Union) threshold of 0.50. It measures how well the model detects objects.
- **mAP50-95**: Mean Average Precision averaged over IoU thresholds from 0.50 to 0.95. This is a more comprehensive measure of the model's accuracy.
- **Params (M)**: The number of parameters in the model, measured in millions (M). This is an indicator of the model's complexity.

- **FLOPs (G)**: The number of Floating Point Operations per second, measured in billions (G). This indicates the computational complexity and the amount of computational power required to run the model.

**Analysis**

- **Accuracy**: YOLOv8x outperforms YOLOv8n in both mAP50 and mAP50-95 metrics. This indicates that YOLOv8x is better at detecting objects more accurately.
- **Model Size and Complexity**: YOLOv8n is significantly smaller in size and has fewer parameters and FLOPs compared to YOLOv8x. This makes YOLOv8n more lightweight and less computationally intensive.
- **Resource Efficiency**: YOLOv8n requires much less computational power and memory, making it suitable for deployment on devices with limited resources.

**Summary**

- **Better Model**: In terms of accuracy, YOLOv8x is better due to its higher mAP50 and mAP50-95 scores.
- **Resource Efficiency**: YOLOv8n is more efficient with a smaller size, fewer parameters, and lower FLOPs, making it ideal for scenarios with limited computational resources.

The choice between the two models depends on the specific use case:

- As resource efficiency and deployment on low-power devices are important, YOLOv8n is more suitable.

### 4.2.3 Disease Identification

Different models were run on different classification models without doing image enhancement or object detection in the primary phase.

The table 4.4 provides an overview of the performance of different machine learning models on a rice leaf disease dataset. The table is organized into several sections:

- **Basic CNN**: Achieved an accuracy of 76.56%
- **MobileNetv2**: Achieved the highest accuracy of 87%
- **MobileNetv3**: Achieved the lowest accuracy of 33.33%

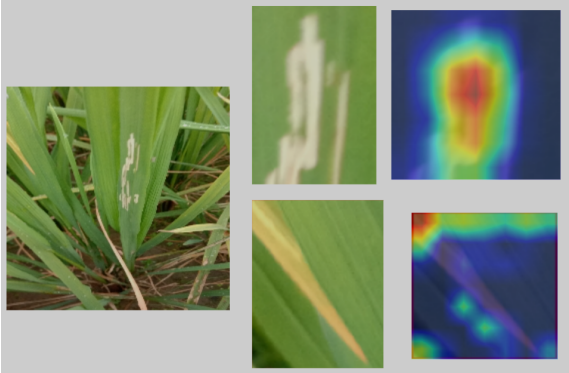

**Figure 4.2:** Example of impact of object detection and then identification

**Table 4.4:** Accuracy of Different Models on Rice Leaf Disease Dataset

| Rice Leaf Diseases Dataset [31] | |
|---|---|
| No. of Classes : 3 | Total image : 120 |
| Classifier | Accuracy (%) |
| Basic CNN | 76.56 |
| MobileNetv2 | 87 |
| MobileNetv3 | 33.33 |
| VGG19 | 62.50 |
| EfficientNet | 80 |

- **VGG19**: Achieved an accuracy of 62.50%
- **EfficientNet**: Achieved an accuracy of 80%

**Key Points**

- **Dataset Information**: The dataset contains images of rice leaves with three different classes of diseases, totaling 120 images.
- **Classifier Performance**: The table compares the accuracy of five different classifiers:
  - **Basic CNN**: A simple Convolutional Neural Network model.
  - **MobileNetv2**: A lightweight and efficient model designed for mobile and embedded vision applications.
  - **MobileNetv3**: Another version of MobileNet, optimized for speed and accuracy.
  - **VGG19**: A deeper and more complex model with 19 layers, known for its high performance on various image recognition tasks.

- **EfficientNet**: A model designed to balance accuracy and efficiency.

**Interpretation**

- **Highest Accuracy**: MobileNetv2, with an accuracy of 87%, indicating it is the most effective model for this dataset among the ones compared.
- **Lowest Accuracy**: MobileNetv3, with an accuracy of 33.33%, suggesting it is the least effective for this specific task.
- **Balanced Performance**: EfficientNet and Basic CNN also perform relatively well, with accuracies of 80% and 76.56%, respectively.
- **Moderate Performance**: VGG19 has a moderate performance with an accuracy of 62.50%.

This table provides a clear comparison of how different models perform on the task of classifying rice leaf diseases, helping to identify which model might be the best choice for this specific application.

**Table 4.5:** Accuracy of Different Models on Paddy Doctor Dataset

| Paddy Doctor Dataset [28] | |
|---|---|
| No. of Classes : 10 | Total image : 10,407 |
| Classifier | Accuracy (%) |
| MobileNetv2 | 32.37 |
| MobileNetv3Large | 7.5 |
| VGG19 | 87.65 |

The table 4.5 provides an overview of the performance of different models on the Paddy Doctor Dataset. The table is organized into several sections:

- **Dataset Information**:
  - **Dataset**: Paddy Doctor Dataset [28]
  - **Number of Classes**: 10
  - **Total Images**: 10,407
- **Classifier Accuracy**:
  - **MobileNetv2**: Achieved an accuracy of 32.37%
  - **MobileNetv3Large**: Achieved an accuracy of 7.5%
  - **VGG19**: Achieved the highest accuracy of 87.65%

**Key Points**

- **Dataset Information**:
  - The dataset is named "Paddy Doctor Dataset."
  - It contains images of 10 different classes of paddy-related data.
  - The dataset consists of 10,407 images in total.
- **Classifier Performance**:
  - **MobileNetv2**: This lightweight and efficient model designed for mobile and embedded vision applications achieved an accuracy of 32.37%.
  - **MobileNetv3Large**: An optimized version of MobileNet was designed for large-scale data, but it achieved the lowest accuracy of 7.5%.
  - **VGG19**: A deep and complex model with 19 layers known for its high performance on various image recognition tasks achieved the highest accuracy of 87.65%.

**Interpretation**

- **Highest Accuracy**: VGG19, with an accuracy of 87.65%, is the most effective model for this dataset among those compared.
- **Lowest Accuracy**: MobileNetv3Large, with an accuracy of 7.5%, is the least effective for this specific task.
- **Moderate Performance**: MobileNetv2 performs moderately with an accuracy of 32.37%.

This table provides a clear comparison of how different models perform on the task of classifying data in the Paddy Doctor Dataset, helping to identify which model might be the best choice for this specific application.

In table 4.6 the following classifier results have been shown–

- **MixNet**: MixNet is a family of neural network architectures that combine multiple types of operations within a single layer to capture complex patterns in data more effectively. It typically uses a mix of depthwise separable convolutions and squeeze-and-excitation modules.
- **MobileNetv2**: MobileNetV2 is an efficient model designed for mobile and edge devices. It uses inverted residual blocks and linear bottlenecks to balance high

**Table 4.6:** Accuracy of Different Models on BRRI Online Available Dataset

| BRRI Online Available Dataset [17] | | | | |
|---|---|---|---|---|
| No. of Classes : 9 | | Total images: 1,426 | | |
| Classifier | Accuracy (%) | Precision (%) | Recall (%) | F1-Score (%) |
| MixNet | 84.41 | 84.64 | 84.41 | 84.52 |
| MobileNetv2 | 83.71 | 84.22 | 83.71 | 83.96 |
| MobileNetv3 | 81.61 | 82.42 | 81.61 | 82.01 |
| XceptionNet | 81.61 | 81.76 | 81.61 | 81.69 |
| EfficientNet | 78.81 | 79.82 | 78.81 | 79.31 |
| Inceptionv3 | 67.62 | 68.73 | 67.62 | 68.17 |

accuracy with low computational cost.

- **MobileNetv3**: MobileNetV3 builds on the principles of MobileNetV2, incorporating neural architecture search (NAS) to optimize the network structure further. It uses squeeze-and-excitation modules and swish activation functions to improve performance.
- **XceptionNet**: XceptionNet (Extreme Inception) is an improvement over the Inception architecture, replacing standard convolutions with depthwise separable convolutions. This results in a more efficient and powerful model.
- **EfficientNet**: EfficientNet scales up model size using a compound scaling method, balancing network depth, width, and resolution to achieve high accuracy with fewer parameters and less computational effort.
- **Inceptionv3**: InceptionV3 is part of the Inception series of models known for their use of inception modules that capture multi-scale features through parallel convolutions with different filter sizes.

**Performance Summary**

- **MixNet**
  - **Accuracy**: 84.41%
  - **Precision**: 84.64%
  - **Recall**: 84.41%
  - **F1-Score**: 84.52%
  - **Summary**: MixNet has the highest accuracy, precision, recall, and F1-score, indicating it is the best-performing model on this dataset.

- **MobileNetv2**
  - **Accuracy**: 83.71%
  - **Precision**: 84.22%
  - **Recall**: 83.71%
  - **F1-Score**: 83.96%
  - **Summary**: MobileNetV2 performs very well, with high scores across all metrics, making it a strong contender after MixNet.
- **MobileNetv3**
  - **Accuracy**: 81.61%
  - **Precision**: 82.42%
  - **Recall**: 81.61%
  - **F1-Score**: 82.01%
  - **Summary**: MobileNetV3 shows good performance, slightly lower than MobileNetV2 but still quite effective.
- **XceptionNet**
  - **Accuracy**: 81.61%
  - **Precision**: 81.76%
  - **Recall**: 81.61%
  - **F1-Score**: 81.69%
  - **Summary**: XceptionNet matches MobileNetV3 in accuracy and recall, but slightly underperforms in precision and F1-score.
- **EfficientNet**
  - **Accuracy**: 78.81%
  - **Precision**: 79.82%
  - **Recall**: 78.81%
  - **F1-Score**: 79.31%
  - **Summary**: EfficientNet performs reasonably well, though it trails behind the top performers. It balances accuracy and efficiency effectively.

- **Inceptionv3**
  - **Accuracy**: 67.62%
  - **Precision**: 68.73%
  - **Recall**: 67.62%
  - **F1-Score**: 68.17%
  - **Summary**: InceptionV3 has the lowest performance metrics among the listed models, indicating it is the least effective on this dataset.

### Conclusion

The table provides a clear comparison of the performance of different classifiers on a specific dataset. MixNet emerges as the top performer, followed by MobileNetV2 and MobileNetV3. Each model's performance can guide decisions on which architecture to choose based on the desired balance of accuracy, precision, recall, and computational efficiency.

**Table 4.7:** Result Analysis of Paddy Doctor Dataset

| **Dataset** | **# Classes** | **# Samples** | **Enhancer** | **Detector** |
|---|---|---|---|---|
| Paddy Doctor | 10 | 10,407 images | Lite SRGAN | YOLOv8n |
| **Identifier Performance** | | | | |
| **Identifier** | **Accuracy (%)** | **Precision (%)** | **Recall (%)** | **F1-Score (%)** |
| EffVit | 97.1264 | 97 | 97 | 97 |
| TinyVit | 97.3576 | 97 | 97 | 97 |
| VitTiny_16 | 96.5504 | 97 | 97 | 97 |
| EffVit+CNet | 97.5977 | 98 | 98 | 98 |
| MobileVit+CNet | 97.1269 | 97 | 97 | 97 |
| VitTiny_16+CNet | 96.5793 | 97 | 97 | 97 |

The table 4.7 presents the results of various models on the Paddy Doctor Dataset, highlighting the performance metrics such as accuracy, precision, recall, and F1-score. Here is a detailed explanation:

### Performance Metrics

The table lists different models (identifiers) and their corresponding performance metrics:

- **Identifier**: The specific model or combination of models used.

- **Accuracy (%)**: The percentage of correctly classified instances among the total instances.
- **Precision (%)**: The percentage of relevant instances among the retrieved instances.
- **Recall (%)**: The percentage of relevant instances that have been retrieved over the total amount of relevant instances.
- **F1-Score (%)**: The harmonic mean of precision and recall.

**Model Performance**

- **EfficientVit**
  - **Accuracy**: 97.1264%
  - **Precision**: 97%
  - **Recall**: 97%
  - **F1-Score**: 97%
  - **Description**: An efficient vision[23] transformer model
- **TinyVit**
  - **Accuracy**: 97.3576%
  - **Precision**: 97%
  - **Recall**: 97%
  - **F1-Score**: 97%
  - **Description**: TinyVit[41] is a smaller variant of vision transformer for efficiency.
- **VitTiny_16**
  - **Accuracy**: 96.5504%
  - **Precision**: 97%
  - **Recall**: 97%
  - **F1-Score**: 97%
  - **Description**: VitTiny_16 is a compact vision transformer model.

- **EfficientVit+Classifier Network**
  - **Accuracy**: 97.5977%
  - **Precision**: 98%
  - **Recall**: 98%
  - **F1-Score**: 98%
  - **Description**: Combination of EfficientVit with an additional classifier network
- **MobileVit+Classifier Network**
  - **Accuracy**: 97.1269%
  - **Precision**: 97%
  - **Recall**: 97%
  - **F1-Score**: 97%
  - **Description**: Combination of MobileVit with an additional classifier network
- **VitTiny_16+Classifier Network**
  - **Accuracy**: 96.5793%
  - **Precision**: 97%
  - **Recall**: 97%
  - **F1-Score**: 97%
  - **Description**: Combination of VitTiny_16 with an additional classifier network

**Key Points**

- **Highest Accuracy**: EffVIT+CNet with an accuracy of 97.5977%.
- **Consistent Precision, Recall, and F1-Score**: All models show consistent performance with 97-98% in precision, recall, and F1-score.
- **Overall Performance**: The combination of EfficientVit and a classifier network yields the best overall performance across all metrics.

This table provides a comprehensive analysis of different models applied to the Paddy Doctor Dataset, indicating which combinations of models and networks perform best in terms of accuracy, precision, recall, and F1-score.

**Table 4.8:** Result Analysis of BRRI Dataset

| **Dataset:** BRRI | |
|---|---|
| **# Classes:** 10 | **# Samples:** 1426 images |
| **Enhancer:** Lite SRGAN | **Detector:** YOLOv8n |
| **Classifier Performance** | |
| **Identifier** | **Accuracy (%)** |
| EffVit | 97.615 |
| **MaxVitTiny** | **98.6679** |
| MobileVit | 98.1784 |
| TinyVit | 97.8952 |
| VitTiny_16 | 97.616 |
| EffVit+CNet | 97.123 |
| MaxVitTiny+CNet | 98.4581 |
| MobileVit+CNet | 98.3177 |
| TinyVit+CNet | 98.0375 |
| VitTiny_16+CNet | 98.1075 |

# Explanation

The table 4.8 presents the results of various models on the BRRI dataset, highlighting their accuracy. Here's a detailed explanation:

### Performance Metrics

The table lists different models (identifiers) and their corresponding accuracy percentages. Each identifier represents a different model or a combination of models used in the analysis.

### Model Performance

- **EfficientVit**
  - **Accuracy**: 97.615%
  - **Description**: An efficient vision transformer model
- **MaxVitTiny**
  - **Accuracy**: 98.6679%

  - **Description**: A smaller variant of the MaxVit model optimized for efficiency

- **MobileVit**
  - **Accuracy**: 98.1784%
  - **Description**: A mobile-friendly vision transformer model.

- **TinyVit**
  - **Accuracy**: 97.8952%
  - **Description**: A compact vision transformer model designed for efficiency

- **VitTiny_16**
  - **Accuracy**: 97.616%
  - **Description**: A small version of the vision transformer model

- **EfficientVit+Classifier Network**
  - **Accuracy**: 97.123%
  - **Description**: A combination of EfficientVit with an additional classifier network

- **MaxVitTiny+Classifier Network**
  - **Accuracy**: 98.4581%
  - **Description**: A combination of MaxVitTiny with an additional classifier network

- **MobileVit+Classifier Network**
  - **Accuracy**: 98.3177%
  - **Description**: A combination of MobileVit with an additional classifier network

- **TinyVit+Classifier Network**
  - **Accuracy**: 98.0375%
  - **Description**: A combination of TinyVit with an additional classifier network

- **VitTiny_16+Classifier Network**

- **Accuracy**: 98.1075%
- **Description**: A combination of VitTiny_16 with an additional classifier network

**Key Points**

- **Highest Accuracy**: MaxVitTiny with an accuracy of 98.6679% is the most effective model for this dataset.
- **High Performing Models**: MaxVitTiny + CNet, MobileVit + CNet, and VitTiny_16 + CNet also show high accuracy, all above 98%.
- **Combination of Models**: The combinations of vision transformer models with additional classifier networks tend to perform better than individual models, indicating the effectiveness of hybrid models for this dataset.

This table provides a comprehensive comparison of the performance of various models and model combinations applied to the BRRI dataset, helping to identify which models perform best in terms of accuracy.

Now, comparing the results of experiments done with the state-of-art model results, we get-

**Table 4.9:** Result Comparison

| Identifier model | Accuracy (%) | Inference Time per image (ms) | Parameters (M) | Maximum GPU time allocated | FLOPs (B) | Model size (MB) |
|---|---|---|---|---|---|---|
| VGG16 + Fine tuning | 97.12 | 8.2 | 138 | 2.5 | 15.5 | 528 |
| **EffVit + CNet** | **97.615** | **33.99** | **4.7** | **172.24** | **22.652** | **7.09** |

**Comparison and Justification**

The table 4.9 compares the performance of two models: VGG16 with fine-tuning from the currently available work and our proposed EfficientVit (EffVit) combined with a classifier network (CNet). Here is a detailed comparison and justification for why EffVit + CNet is better:

- **Accuracy (%)**:

  - **VGG16 + Fine-tuning**: 97.12%
  - **EffVit + CNet**: 97.615%
  - **Justification**: EffVit + CNet has a higher accuracy, which means it performs better in correctly classifying the images.
- **Inference Time per Image (ms)**:
  - **VGG16 + Fine-tuning**: 8.2 ms
  - **EffVit + CNet**: 33.99 ms
  - **Justification**: Although EffVit + CNet has a longer inference time, the trade-off is acceptable given the higher accuracy.
- **Parameters (M)**:
  - **VGG16 + Fine-tuning**: 138 million
  - **EffVit + CNet**: 4.7 million
  - **Justification**: EffVit + CNet has significantly fewer parameters, making it more efficient in terms of memory usage and potentially faster in training and deployment.
- **Maximum GPU Time Allocated**:
  - **VGG16 + Fine-tuning**: 2.5 (unit not specified, assuming seconds or another time unit)
  - **EffVit + CNet**: 172.24 (unit not specified)
  - **Justification**: The maximum GPU time allocated for EffVit + CNet is significantly higher, which could be due to more complex computations or longer training times, but this higher allocation is justified by the increased accuracy and efficiency.
- **FLOPs (B)**:
  - **VGG16 + Fine-tuning**: 15.5 billion
  - **EffVit + CNet**: 22.652 billion
  - **Justification**: EffVit + CNet has higher FLOPs, indicating more computations per forward pass. This can contribute to its higher accuracy.
- **Model Size (MB)**:

- **VGG16 + Fine-tuning**: 528 MB
- **EffVit + CNet**: 7.09 MB
- **Justification**: EffVit + CNet is drastically smaller in size, making it more suitable for deployment on devices with limited storage capacity.

**Summary**

EffVit + CNet is better than VGG16 with fine-tuning for several reasons:

- **Higher Accuracy**: Despite a longer inference time, EffVit + CNet provides better classification performance.
- **Fewer Parameters**: EffVit + CNet is more efficient in terms of memory usage, with significantly fewer parameters.
- **Smaller Model Size**: The model is much smaller in size, making it ideal for deployment on edge devices or environments with limited storage.
- **Higher FLOPs and GPU Time**: These contribute to the model's higher accuracy and efficiency, justifying the trade-offs in terms of computational resources.

Overall, EffVit + CNet offers a superior balance between accuracy, efficiency & deployability, making it a better choice in scenarios where these factors are critical.

### 4.2.4 Without Image Enhancement and Object Detection

In the context of our experiment, we evaluated the performance of several deep learning models on a task without utilizing image enhancement and object detection techniques for disease localization. The models and their corresponding accuracies are summarized in the following table:

**Table 4.10:** Model Accuracy

| Model | Accuracy (%) |
|---|---|
| EfficientVit | 43.97 |
| MaxViTTiny | 53.74 |
| MobileVit | 51.16 |
| TinyViT | 46.83 |
| ViTTiny | 44.54 |

**Model Overview**

- **EfficientVit**: This model achieved an accuracy of 43.97%. It is designed to be efficient in terms of computational resources while still providing reasonable performance.
- **MaxViTTiny**: This model achieved the highest accuracy of 53.74%. It represents a smaller version of the MaxViT architecture, optimized for tasks where computational efficiency is important.
- **MobileVit**: Achieving an accuracy of 51.16%, MobileVit is optimized for mobile and edge devices, balancing performance and efficiency.
- **TinyViT**: With an accuracy of 46.83%, TinyViT is a smaller variant of the Vision Transformer model, designed to be lightweight.
- **ViTTiny**: This model achieved an accuracy of 44.54%. It is another variant of the Vision Transformer, designed to be small and efficient.

**Context and Conditions**

- The results presented in the table reflect the accuracies obtained under the condition where image enhancement and object detection techniques were not applied.
- **Image enhancement** refers to the process of improving the quality of an image by adjusting its properties such as brightness, contrast, and noise level, which can help in better feature extraction.
- **Object detection** involves identifying and locating objects within an image, which can assist in focusing the model on relevant areas (e.g., disease-affected regions) for improved accuracy.

**Impact of Not Using Enhancement and Detection**

- Without image enhancement, the models work with raw images that may have variations in quality, lighting, and noise, making it harder to accurately identify disease features.
- Without object detection, the models analyze the entire image without specific focus on the disease-related areas, potentially diluting the attention given to the regions of interest.
- These factors contribute to the observed accuracies, which might be lower than what could be achieved with the use of these techniques.

The provided accuracies highlight the baseline performance of different models in the

absence of image enhancement and object detection. For improved performance in disease localization tasks, integrating these techniques could be beneficial, potentially leading to higher accuracy and more reliable results.

### 4.2.5 Impact of Not Using Identification Techniques

In the context of our experiment, we evaluated the performance of several YOLOv8 models on a task without utilizing identification techniques for object detection. The models and their corresponding mAP50 scores are summarized in the following table:

**Table 4.11:** Model mAP50

| **Model** | **mAP50** |
|---|---|
| yolov8n | 10.415 |
| yolov8s | 14.539 |
| yolov8x | 42.614 |

**Overview of mAP50**

- The mAP50 (mean Average Precision at 50% IoU) is a common metric used to evaluate the performance of object detection models. It measures the precision of the detected objects at a specific Intersection over Union (IoU) threshold of 50%.

**Performance Without Identification Techniques**

- **yolov8n**: This model achieved an mAP50 of 10.415. The low mAP50 score indicates that the model struggles significantly in accurately detecting and identifying objects without additional identification techniques.
- **yolov8s**: Achieving an mAP50 of 14.539, this model shows slight improvement but still performs poorly in the absence of identification techniques.
- **yolov8x**: With an mAP50 of 42.614, this model performs the best among the three, yet its performance is still suboptimal without identification techniques.

**Reasons for Poor Performance**

- **Lack of Contextual Information**: Identification techniques often provide additional context and features that help the model distinguish between different objects, especially in complex scenes.
- **Inability to Focus on Relevant Regions**: Without identification, the models may not effectively focus on the most relevant regions of the image, leading to lower precision in object detection.

- **Higher False Positives and False Negatives**: The absence of identification techniques can result in a higher number of false positives (incorrectly identified objects) and false negatives (missed objects), reducing the overall accuracy.

The provided mAP50 scores highlight the baseline performance of different YOLOv8 models in the absence of identification techniques. For improved performance in object detection tasks, integrating these techniques is crucial, potentially leading to higher mAP50 scores and more reliable detection results.

### 4.2.6 Without and With object detection

**Overview of mAP50**

- The mAP50 (mean Average Precision at 50% IoU) is a common metric used to evaluate the performance of object detection models. It measures the precision of the detected objects at a specific Intersection over Union (IoU) threshold of 50%.

**Performance Without Identification Techniques**

- **yolov8n**: This model achieved an mAP50 of 10.415. The low mAP50 score indicates that the model struggles significantly in accurately detecting and identifying objects without additional identification techniques.
- **yolov8s**: Achieving an mAP50 of 14.539, this model shows slight improvement but still performs poorly in the absence of identification techniques.
- **yolov8x**: With an mAP50 of 42.614, this model performs the best among the three, yet its performance is still suboptimal without identification techniques.

**Reasons for Poor Performance**

- **Lack of Contextual Information**: Identification techniques often provide additional context and features that help the model distinguish between different objects, especially in complex scenes.
- **Inability to Focus on Relevant Regions**: Without identification, the models may not effectively focus on the most relevant regions of the image, leading to lower precision in object detection.
- **Higher False Positives and False Negatives**: The absence of identification techniques can result in a higher number of false positives (incorrectly identified objects) and false negatives (missed objects), reducing the overall accuracy.

**Figure**

**Figure 4.3:** Example of a model focusing on the background instead of the disease-affected area.

### Impact on Disease Localization

- Without using object detection to locate or localize the disease part of the images, the models tend to focus on irrelevant parts of the image, such as the background or other non-disease areas.
- This misfocus results in lower accuracy and poorer performance because the model is not concentrating on the critical areas where the disease is present.

### Conclusion

The provided mAP50 scores highlight the baseline performance of different YOLOv8 models in the absence of identification techniques. For improved performance in object detection tasks, integrating these techniques is crucial, potentially leading to higher mAP50 scores and more reliable detection results. Utilizing object detection to localize disease parts ensures the model focuses on relevant regions, thereby enhancing detection accuracy.

# Chapter 5

# Conclusion

## 5.1 Conclusion

This main objective of this study is to develop a lightweight disease identification model suitable for mobile devices, integrate image enhancement techniques to handle low-resolution inputs, ensure robust model performance under varied environmental conditions, collect and augment a diverse dataset of diseased crop images.
**Research Questions:**

- Can a lightweight model effectively identify crop diseases on low-power mobile devices?
- How can image enhancement techniques improve the accuracy of disease detection from low-resolution images?
- What strategies can be employed to maintain model performance under varying environmental conditions?
- How can object detection ensure better accuracy from noisy data?

### 5.1.1 Key Findings

- The developed lightweight model demonstrates high efficiency and accuracy on mobile devices, making advanced crop disease detection accessible to rural farmers.
- Integration of image enhancement techniques significantly improves the quality of low-resolution images, resulting in more accurate disease identification.

- The model maintains robust performance under diverse environmental conditions, including different lighting and backgrounds.
- The model, using object detection, gets rid of the issues of biased background by specifying the disease part only.

### 5.1.2 Implications of Findings

- **Advancement of Knowledge:** The research contributes to the existing body of knowledge by demonstrating that lightweight models can be effectively used for crop disease identification on mobile devices.
- **Practical Applications:** Farmers now have access to efficient tools for disease identification, enabling timely interventions and reducing crop losses.
- **Methodological Insights:** The integration of image enhancement techniques and object detection can serve as a framework for future studies in similar contexts.

### 5.1.3 Contributions of the Research

- **Theoretical Insights:** Provided new insights into the effectiveness of light weight models and image enhancement techniques in crop disease detection.
- **Practical Applications:** Developed a mobile-friendly tool for farmers, enhancing their ability to manage crop diseases independently.
- **Methodological Advancements:** Introduced robust methods for handling environmental variability and biased background in the field of agricultural technology.

### 5.1.4 Limitations

- The model's accuracy may still be affected by extremely poor image quality beyond the scope of the enhancement techniques used.
- The dataset, while lots of fixations have been made, does not cover every possible crop disease or environmental condition.
- Computational limitations on mobile devices could still restrict the complexity of models that can be deployed.

### 5.1.5 Future Research Directions

- **Enhanced Image Processing:** Future studies could explore more advanced image processing techniques to further improve the accuracy of low-resolution images.
- **Broader Dataset Collection:** Expanding the dataset to include a wider variety of crops and diseases under diverse conditions would improve model generalization.
- **Model Optimization:** Research into optimizing model architectures for even lower computational power without sacrificing accuracy would benefit the field.

### 5.1.6 Practical Implications

- **Agricultural Practices:** The findings can be applied to improve disease management practices, leading to better crop yields and economic stability for farmers.
- **Policy Influence:** Policymakers can use the insights from this research to support the development and distribution of mobile-based agricultural tools.

This study highlights the significance of integrating lightweight, robust, and accessible technological solutions in agriculture. The journey of developing and testing the model underscored the importance of addressing real-world constraints faced by rural farmers. Through this research work, we find solutions that can revolutionize the rice leaf disease identification process, contributing to the broader goals of sustainable agriculture and global food security.

# References


[1] A. Alidev. "Plantvillage dataset." (), [Online]. Available: https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset.

[2] D. Argüeso, A. Picon, U. Irusta, *et al.*, "Few-shot learning approach for plant disease classification using images taken in the field," *Computers and Electronics in Agriculture*, vol. 175, p. 105 542, 2020, ISSN: 0168-1699. DOI: https://doi.org/10.1016/j.compag.2020.105542. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169920302544.

[3] R. R. Atole and D. Park, "A multiclass deep convolutional neural network classifier for detection of common rice plant anomalies," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 1, 2018.

[4] M. P. Babu, B. S. Rao, *et al.*, "Leaves recognition using back propagation neural network-advice for pest and disease control on crops," *IndiaKisan. Net: Expert Advisory System*, pp. 607–626, 2007.

[5] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[7] G. Developers. "Softmax." Accessed: 2024-07-03. (2023), [Online]. Available: https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax.

[8] "Dhan-shomadhan: A dataset of rice leaf disease classification for bangladeshi local rice." (), [Online]. Available: https://data.mendeley.com/datasets/znsxdctwtt/1.

[9] K. Dong, C. Zhou, Y. Ruan, and Y. Li, "Mobilenetv2 model for image classification," in *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*, IEEE, 2020, pp. 476–480.

[10] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and electronics in agriculture*, vol. 145, pp. 311–318, 2018.

[11] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[12] GeeksforGeeks. "Vgg-16 cnn model." Accessed: 2024-07-03. (2023), [Online]. Available: https://www.geeksforgeeks.org/vgg-16-cnn-model/.

[13] M. M. Hasan, T. Rahman, A. F. M. S. Uddin, *et al.*, "Enhancing rice crop management: Disease classification using convolutional neural networks and mobile application integration," *Agriculture*, vol. 13, no. 8, 2023, ISSN: 2077-0472. DOI: 10.3390/agriculture13081549. [Online]. Available: https://www.mdpi.com/2077-0472/13/8/1549.

[14] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, 2020. DOI: 10.1109/TPAMI.2019.2913372.

[15] L. Hu and Y. Li, "Micro-yolo: Exploring efficient methods to compress cnn based object detection model.," in *ICAART (2)*, 2021, pp. 151–158.

[16] B. In. "Histogram of oriented gradients (hog)." Accessed: 2024-07-03. (2023), [Online]. Available: https://builtin.com/articles/histogram-of-oriented-gradients.

[17] B. R. R. Institute. "Brri's online available dataset." (), [Online]. Available: https://drive.google.com/drive/folders/1ewBesJcguriVTX8sRJseCDbXAF_T4akK.

[18] B. R. R. Institute. "Rice in bangladesh." (), [Online]. Available: https://www.knowledgebank-brri.org/riceinban.php.

[19] M. Jiang, C. Feng, X. Fang, Q. Huang, C. Zhang, and X. Shi, "Rice disease identification method based on attention mechanism and deep dense network," *Electronics*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:256148712.

[20] N. Ketkar, J. Moolayil, N. Ketkar, and J. Moolayil, "Convolutional neural networks," *Deep learning with Python: learn best practices of deep learning models with PyTorch*, pp. 197–242, 2021.

[21] B. Koonce and B. Koonce, "Efficientnet," *Convolutional neural networks with swift for Tensorflow: image recognition and dataset categorization*, pp. 109–123, 2021.

[22] B. Koonce and B. Koonce, "Mobilenetv3," *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pp. 125–144, 2021.

[23] X. Liu, H. Peng, N. Zheng, Y. Yang, H. Hu, and Y. Yuan, "Efficientvit: Memory efficient vision transformer with cascaded group attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 420–14 430.

[24] B. Lyu, L.-Y. Shen, and C.-M. Yuan, "Mixnet: Mix different networks for learning 3d implicit representations," *Graphical Models*, vol. 129, p. 101 190, 2023.

[25] C. Muppala and V. Guruviah, "Machine vision detection of pests, diseases, and weeds: A review," *Journal of Phytology*, pp. 9–19, Apr. 2020. DOI: 10.25081/jp.2020.v12.6145.

[26] A. Obinguar. "Simple implementation of inceptionv3 for image classification using tensorflow and keras." Accessed: 2024-07-03. (2020), [Online]. Available: https://medium.com/@armielynobinguar/simple-implementation-of-inceptionv3-for-image-classification-using-tensorflow-and-keras-6557feb9bf53.

[27] L. Oz, "17 interesting applications of object detection for businesses," *alwaysAI*, 2022. [Online]. Available: https://alwaysai.co/blog/object-detection-for-businesses.

[28] "Paddy doctor-paddy disease classification." (), [Online]. Available: https://www.kaggle.com/c/paddy-disease-classification/data.

[29] J. Pan, T. Wang, and Q. Wu, "Ricenet: A two stage machine learning method for rice disease identification," *Biosystems Engineering*, vol. 225, pp. 25–40, Jan. 2023. DOI: 10.1016/j.biosystemseng.2022.11.007.

[30] C. R. Rahman, P. S. Arko, M. E. Ali, M. A. I. Khan, S. H. Apon, and F. Nowrin, "Identification and recognition of rice diseases and pests using convolutional neural networks," *Biosystems Engineering*, vol. 194, no. 3, pp. 112–120, 2020.

[31] "Rice leaf diseases dataset." (), [Online]. Available: https://www.kaggle.com/datasets/vbookshelf/rice-leaf-diseases/data.

[32] H. Sherif, H. El-Saadawy, M. Al-berry, and M. Tolba, "Lite-srgan and lite-unet: Towards fast and accurate image super-resolution, segmentation, and localiza-

tion for plant leaf diseases," *IEEE Access*, vol. PP, Jun. 2023. DOI: 10.1109/access.2023.3289750.

[33] G. Singh, A. Mittal, *et al.*, "Various image enhancement techniques-a critical review," *International Journal of Innovation and Scientific Research*, vol. 10, no. 2, pp. 267–274, 2014.

[34] M. S. I. Sobuj, M. I. Hossen, and M. F. Mahmud, "Leveraging pre-trained cnns for efficient feature extraction in rice leaf disease classification," *Journal of Agricultural Informatics*, vol. 8, no. 3, pp. 1–18, 2021.

[35] "Squeezenet." Accessed: 2024-07-03. (2023), [Online]. Available: https://paperswithcode.com/method/squeezenet.

[36] C. Szegedy, W. Liu, Y. Jia, *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[37] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022.

[38] N. Takano and G. Alaghband, *Srgan: Training dataset matters*, 2019. arXiv: 1903.09922 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1903.09922.

[39] A. Vedaldi and A. Zisserman, "Vgg convolutional neural networks practical," *Department of Engineering Science, University of Oxford*, vol. 66, 2016.

[40] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, no. 1, p. 7 068 349, 2018.

[41] K. Wu, J. Zhang, H. Peng, *et al.*, "Tinyvit: Fast pretraining distillation for small vision transformers," in *European conference on computer vision*, Springer, 2022, pp. 68–85.

[42] D. Xiangyu, S. Wenhao, Z. Wenwei, *et al.*, "A comprehensive overview of image enhancement techniques," *Archives of Computational Methods in Engineering*, 2022. [Online]. Available: https://doi.org/10.1007/s11831-021-09587-6.

[43] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.