Is Image Segmentation all we need?

Saim Ur Rehman **Jawad Shahid**
l215242@lhr.nu.edu.pk l215787@lhr.nu.edu.pk
Department of Computer Science (CS)
National University of Computer and Emerging Sciences, Lahore, Pakistan

Abstract

Ever since the advent of AI in agriculture, the demand for highly accurate classifiers is growing exponentially and one way to achieve this goal is to feed in high-grade data to the models. We, therefore, present two novel approaches for segmenting the areas of interest, hence, removing the noise from images of infected rice leaves, and by keeping the models constant, provide a detailed comparison between the two approaches, along with the unsegmented data. The comparison is based upon five different datasets comprising of several rice leaf diseases including Rice Blast, Rice Hispa, Bacterial leaf Blight, Sheath Blight, Rice Turgro, Brown Spot and Narrow Brown Spot. For testing, we use seven different pretrained models, and conclude that segmentation shows a considerable increase in accuracy compared to unsegmented data. The first approach uses a combination of traditional segmentation algorithms i.e. Canny Edge Detection and Otsu thresholding, while the second approach uses a combination of the SOTA models i.e. YOLO Object detector and SAM-2 segmentation model to achieve the desired outcome. Results show impressive increases in classifier accuracy when the data is trained on the refined dataset, which is produced using the first segmentation approach. On the other hand, when the second approach is combined with Reduplication augmentation, it provides even better results in increasing accuracy of the classifiers. Both of these approaches are compared for their strengths and weaknesses. Lastly, we also utilize GAN to generate synthetic data and combine it with our segmentation approach, which enhances the accuracy by a huge factor, to show the impact of reducing the lack of data.

1 Introduction

Rice is a primary staple crop which feeds over half of the global population. It directly impacts food security in some countries where it is a core part of diet. At the same time, common diseases affecting rice leaves, such as bacterial blight, rice blast, and brown spot, pose severe threats to crop yields and quality. Rice crops affected by the rice blast disease may result in a 10 to 20% decrease in yield, annually (1). Traditionally, these diseases can be identified visually or through laboratory tests. Both of these methods require an expert input, and are time-intensive. For this reason, there is an urgent need for automated and accurate methods for early rice disease detection.

The use of machine learning and deep learning in this field has shown promise in enhancing disease identification processes by allowing for highly accurate and automated analyses (2; 3; 4). Specifically, convolutional neural networks (CNNs) have become a popular choice in image-based disease detection tasks due to their strong feature extraction capability. Despite this, model training from scratch, as with the use of a CNN, requires extensive datasets. These datasets must be large, and are difficult to obtain in agricultural contexts (2).

Alternatively, transfer learning is a technique which lessens the scarcity of the data, as it uses pre-trained models. These models are developed on large-scale datasets, and can be adapted to new tasks with minimal data. In recent years, models such as DenseNet, EfficientNet, and ResNet, have been repurposed for agriculture disease detection tasks. These models have shown to achieve high accuracy, and reduce training time (5). In this paper, we specifically focus on the enhancement of transfer learning models on rice disease detection applications, using segmentation and augmentation techniques which aim to improve disease detection accuracy by utilizing noise reduction, data refinement, and target area isolation.

In plant-based disease detection, accurate segmentation can isolate disease-affected regions(6; 7; 8; 9). Particularly in the case of rice, the dis-

eases are primarily focused on the leaf of the rice plant (3; 10). Using segmentation, we can isolate this leaf, which could enhance the focus of the model, and potentially increase classification accuracy. For this, we have proposed a segmentation algorithm which is a combination of canny edge detection, contour detection, and otsu thresholding. By using this combination, we aim to optimize the input for models being trained on rice disease data, specifically for the following models: InceptionResNetV2, DensNet201, MobileNetV2, ResNet152V2, and EfficientNetB3. We also perform experiments on this approach by using GAN to generate synthetic data based on the segmented datasets to see the effect of augmentation on model accuracy.

Furthermore, we also introduce an approach that makes use of annotated data to identify diseased leaves using the YOLO model, and segments the diseased area using the SAM model. Although this approach incorporates additional steps, it is a highly established method for successfully segmenting rice leaves. We have also performed augmentation on the segmented data produced by this approach. This produces an augmented dataset where each image is filled with specifically the diseased leaf.

This study introduces the novel approaches that combine our custom segmentation techniques and augmentation techniques with transfer learning for rice disease detection. The aim of this is to provide a cleaner input for the models to learn disease specific data. The contributions of this paper are threefold:

1. We propose two segmentation and augmentation pipelines to enhance input quality for rice disease identification.

2. We evaluate the effectiveness of multiple transfer learning models on the segmented and augmented datasets.

3. We analyze model performance to identify the optimal combination of segmentation, augmentation, and classification techniques for accurate rice disease detection.

The rest of this paper is organized as follows: Section 2 reviews related work and highlights the research gap. Section 3 details and describes the datasets used for this study. Section 4 details the proposed methodology, which includes the segmentation, augmentation, and model training processes, as well as experiments. Section 5 presents the results of the experimentation, followed by Section 6 which presents a discussion of the findings. Finally, Section 7 concludes the study, and suggests directions for future research.

2 Related Work

(11) introduces a segmentation method for detecting diseased leaves in kholrabi crops using color negative indices (VIs) and Otsu' thresholding on aerial RGB images. The approach, which is tested under different sunlight conditions, showed that tuning VIs (especially ExGR) improves the accuracy of distinguishing healthy leaves, diseased leaves, and background elements. The study shows that real-time segmentation also performed well with RGB cameras, which are much more cost-effective as compared to multispectral setups. The results suggest that this method is not only affordable, but also useful for high-precision disease detection in agriculture.

(12) addresses the need for portable, accurate crop disease diagnosis by employing a modified LeNet-5 CNN model for classification, and Otsu multi-thresholding for image segmentation. The model is trained on the Plant Village dataset which contains tomato leaf images, and achieves high accuracy in identifying various diseases. Real-time diseased images were also tested to verify the model's generalization capabilities on new data. The study highlights the value brought by Otsu thresholding, as it improves the model's performance. The study also highlights the superior performance of the model compared to other models like Xception, ResNet50, and VGG16, as it demonstrated higher precision, recall, F1-score, and accuracy results. This presented the study as a highly effective solution for practical crop disease detection.

(13) proposes an automatic rice leaf disease segmentation method using image processing techniques. The method involves four main phases: RGB to HSV color transformation, and image segmentation using k-means clustering. Through HSV color transformation, the hue component assists in isolation the disease spots, while the k-means clustering technique effectively partitions the images into groups. This helps to identify the infected regions. The results demonstrate that the method achieves full automation with good effi-

ciency, successfully segmenting disease areas in rice leaf images.

(14) introduces a method for segmenting disease symptoms in plant leaf images. It proposes an algorithm which can effectively classify diseased tissue in plant images by analyzing pixel patterns in under 2 seconds, unlike manual methods which require an average of 66 seconds. It is designed to capture ambiguous regions (fuzzy zones) as diseased, and marks 59-98% of these pixels correctly. Compared to other automatic methods, the algorithm demonstrates lower error rates and a consistently low false positive rate (0.3-6%), achieving an Area Under Curve (AUC) above 0.95 across all disease symptoms, which indicates reliable detection across diverse categories. These results make it a viable and efficient alternative to manual and other automated methods for plant disease assessment.

[15] uses a similar approach to our proposed methodology, except that they did not use reduplication and background retention strategies and, in addition, only focused on medical imaging segmentation. The authors tried several evaluation metrics including F1-score, precision, recall etc. and performed an in-depth analysis using different segmentation models thereby reaching the conclusion that segmentation improves overall accuracy by a huge factor but is subjective to which models are used.

[16] presents several experiments to compare manual segmentation versus segmentation using the SAM model in the field of agriculture. This paper evaluates SAM in several different scenarios to assess whether pretrained segmentation is sufficient for segmenting objects of interest or does manual segmentation have to play a part. The results conclude that the SAM model, although not as accurate as manual segmentation, performs well in the field of agriculture for segmenting different entities and therefore the labor-intensive task of manual segmentation is not necessary, hence saving a significant amount of time.

[17] presents a novel approach using GANs as the base for the creation of synthetic data for rice leaf diseases. This paper elaborates the problems with data imbalance and relatively scarce data and then introduces a GAN based method to increase the dataset by a huge factor while improving the variance factor. The authors then present a detailed analysis comparing results with and without the use of synthetic GAN generated data and conclude that the latter yields a significant increase in accuracy.

[18] performs a survey for several image processing techniques including noise reduction, color extraction, K-means clustering etc. and provides a detailed comparison between these using classification accuracy as the evaluation metric. Further the paper explains advantages and limitations of each technique along with their use-case scenarios. In conclusion, the results demonstrate that given the right scenario for the specific augmentation, a huge accuracy improvement is shown.

3 Datasets

In this study, we utilized five different disease datasets to train and evaluate our models. These datasets, obtained from Mendeley and Kaggle, contain a diverse range of images, each labeled with specific disease types that affect rice plants. Each dataset consists of labelled data, where each class is placed into a separate folder. Table 1 contains a summary of the datasets.

Below are details of each dataset:

3.1 D1: Rice Leaf Disease Image Samples

The D1 dataset from (TODO:D1citation) contains a total of 5932 images, covering four types of rice leaf diseases: Bacterial Blight, Rice Blast, Brown Spot, and Tungro. It contains close-up images of the disease on the rice leaf, for each type of disease. These diseases are common in rice plants and can significantly affect crop yields. The dataset was primarily created and designed to assist with rice leaf disease identification using machine learning techniques. The dataset has been widely used for research on disease classification and detection. The images in this dataset are high-resolution and have been pre-labeled with the appropriate disease categories, making it ideal for both training and evaluation in classification tasks.

3.2 D2: Rice Leaf Diseases Dataset

The D2 dataset sources from (TODO:D2citation) is an extensive collection of 4684 images, focusing on three rice leaf diseases: Bacterial Blight, Brown Spot, and Leaf Smut. In addition to disease labels, the dataset also provides supplementary data, including symptom descriptions and environmental parameters such as temperature and

humidity, which are crucial for understanding disease progression and environmental factors that may contribute to disease spread. The dataset consists of close-up shots of leaves, however, with less noise as compared to D1 as the background is blurred.

3.3 D3: Dhan-Shomadhan Dataset (Field Background)

The D3 dataset from (TODO:D3citation) is part of the Dhan-Shomadhan collection, consisting of 337 images with field backgrounds and representing five different harmful rice leaf diseases: Brown Spot, Leaf Scald, Rice Blast, Rice Tungro, and Sheath Blight. This dataset includes images taken in a natural field setting, with varying lighting and background conditions. The images are much higher resolution than D1 and D2, and the images are fairly distanced, as there is a significant amount of background scenery in the image, as well as images of the leaves themselves. It is suitable for real-world applications where rice leaves are observed under field conditions.

3.4 D4: Dhan-Shomadhan Dataset (White Background)

Similar to D3, the D4 dataset is sourced from (TODO:D4citation). It contains 769 rice leaf images against a white background. The view distance is the same as in D3. The dataset focuses on the same five rice leaf diseases: Brown Spot, Leaf Scald, Rice Blast, Rice Tungro, and Sheath Blight. The white background makes it easier for image processing tasks like segmentation and classification, as there are fewer variables to account for, compared to field images. This dataset is ideal for training and validating models on controlled, high-contrast backgrounds.

3.5 D5: Bangladesh Rice Leaf Disease Dataset

The D5 dataset sourced from (TODO:D5citation) consists of 1701 images representing eight different rice leaf diseases: Bacterial Leaf Blight, Brown Spot, Leaf Scald, Narrow Brown Leaf Spot, Rice Hispa, Sheath Blight, Leaf Blast, and Healthy Rice Leaf. The view distance from the camera for this dataset is the same as for D3 and D4, where a significant amount of background is visible for each image. The dataset was collected from various regions in Bangladesh and contains high-quality images captured under natural conditions. Each disease is represented by a separate class, making it suitable for multi-class classification tasks.

4 Methodologies

In this section, we will detail the methodologies we implemented to attempt to enhance the accuracy of rice disease detection. These include two distinct pipelines with two different segmentation techniques, and two different augmentation techniques. Both these pipelines were compared by the use of several transfer learning classifiers. These transfer learning classifiers were executed with the same parameters on the data produced by each algorithm, as well as the original data for comparison.

The segmentation algorithms aim to isolate the rice leaf in every image in the datasets, which would ideally increase the accuracy of disease detection, as the target diseases exist on the leaf of the rice plant. The goal of this attempt is to reduce noise picked up by the classifier, which would theoretically increase the capability of the model in being able to detect the disease.

The augmentation techniques aim to provide the models with more of the data they need to target, learn, and identify. The augmentation works well on top of segmentation, as the segmentation ideally separates a single diseased leaf and removes the background. Augmentation is then able to multiply this target area for the model.

The methods covered here include a state-of-the-art segmentation pipeline, which combines the SAM-2 and YOLO model, and a custom segmentation approach using Canny Edge Detection & Otsu Thresholding. The augmentation methods include Generative Adverserial Networks (GANs) for synthetic data generation, and Reduplication with background retention for data augmentation. A dedicated ”Classifier” subsection outlines the various transfer learning models used.

4.1 Canny Edge Otsu Segmentation Algorithm

The proposed segmentation algorithm over the state-of-the-art YOLO + SAM-2 approach is the Canny Edge Otsu Segmentation Algorithm. This approach intregrates multiple computer vision techniques, each chosen for its role in enhancing image clarify and emphasizing focus on the leaf

Table 1: Datasets Summary

Dataset Number	Number of Images	Classes
D1	5932	(4) Bacterial Blight, Rice Blast, Brown Spot, Tungro
D2	4684	(3) Brown Spot, Leaf Smut, Bacterial Blight
D3	337	(5) Brown Spot, Leaf Scald, Rice Blast, Rice Tungro, Sheath Blight
D4	769	(5) Brown Spot, Leaf Scald, Rice Blast, Rice Tungro, Sheath Blight
D5	1701	(8) Bacterial Leaf Blight, Brown Spot, Leaf Scald, Narrow Brown Leaf Spot, Rice Hispa, Sheath Blight, Leaf Blast, Healthy Rice Leaf

structure.

The algorithm begins by converting the image to gray-scale. This process simplifies the image by removing color information, which is less crucial for edge detection and segmentation.

$$I_{gray} = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

where:
- I_{gray}: Intensity of the grayscale image.
- R, G, and B: Red, green, and blue intensity values of the original image pixels.

Gray-scale conversion helps reduce the computational requirements by focusing solely on intensity values (light vs. dark regions) rather than color variations.

A Gaussian Blur filter is then applied to the gray scale image to reduce noise.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

where:
- $G(x, y)$: Resulting blurred pixel at coordinates (x, y).
- σ: Standard deviation of the Gaussian distribution, controlling the blur's intensity.
- x and y: Horizontal and vertical distances from the pixel center.

Noise is any unwanted variation in pixel intensity, that can conceal the edges. Gaussian blur smooths the image by averaging the pixel values in a way that emphasizes the central pixels, and reduces sudden intensity changes. This process is achieved using a Gaussian function, which weighs neighboring pixels based on their distance from the central pixel. This creates a softening effect which retains the critical edges.

Otsu's threshold is then used after the gaussian blur to separate the leaf from the background by identifying an optimal threshold level for the gray-scale intensity values.

$$\text{Threshold} = \arg\min_{\theta} \left(\sigma_b^2\right)$$

where:
- θ: Threshold value for distinguishing foreground and background.
- σ_b^2: Variance between the two classes (foreground and background).

Otsu's method automatically calculates this threshold by minimizing the intra-class variance, which makes it ideal for high-contrast areas. Since leaves are typically darker than the background due to shadowing from sunlight, this technique effectively distinguishes the leaf region as a darker segment against a lighter background.

The algorithm then uses a morphological closing operation to further refine the segmented leaf. These operations are often used in image processing to modify the shape of objects in bi-

nary images. The closing operation involves two fundamental steps:

- Dilation: Expands the white regions (in this case, the segmented leaf).
- Erosion: Reduces small noise areas and smooths object boundaries.

$$\text{Closing}(I) = \text{Erosion}(\text{Dilation}(I, K), K)$$

where:
- I: Binary image after thresholding.
- K: Structuring element (e.g., a 4×4 kernel) defining the area used for dilation and erosion.

This operation helps to close any small gaps or holes within the leaf region, making it more cohesive and easier to identify.

Contour detection is used to identify and outline each separate shape within the image, once a clear binary image of the segmented leaf has been obtained.

$$M(x, y) = \begin{cases} 255 & \text{if } (x, y) \in \text{largest contour} \\ 0 & \text{otherwise} \end{cases}$$

where:
- $M(x, y)$: Mask value at pixel (x, y), with 255 representing the leaf area and 0 representing the background.
- (x, y): Pixel coordinates.

Contours are continuous lines or cuves that define the boundaries of objects. The algorithm makes an assumption that the largest contour corresponds to the main leaf in the image, and creates a mask around this shape.

By applying a bitwise operation, the algorithm isolates the leaf from the original image by using the largest contour as a mask, which effectively "cuts out" the leaf area. This mask eliminates surrounding background noise, and allows the model to focus specifically on the leaf.

As a final and additional step, the isolated leaf image is converted to the HSV (Hue, Saturation, Value) color space.

$$G(x, y) = \begin{cases} 255 & \text{if pixel is within green range} \\ 0 & \text{otherwise} \end{cases}$$

where:
- $G(x, y)$: Binary mask for green pixels at location (x, y), where 255 indicates green pixels and 0 indicates non-green pixels.
- HSV range for green: $(35 \leq H \leq 85, 40 \leq S \leq 255, 40 \leq V \leq 255)$.
- H, S, and V: Hue, Saturation, and Value in HSV space.

In HSV, it is easier to separate colors based on their perceived shade value. The algorithm places a filter on the resultant image by defining a range of green hues. If the image does not contain the defined amount of green hue, it is considered to have failed the segmentation process (meaning, the segmented image does not contain a leaf), and it can be safely removed from the dataset. This ensures that the produced dataset is not corrupted, as only images containing a sufficient amount of green pixels are saved for further analysis, ensuring accurate leaf detection.

This approach has several benefits:

- **Reduced Computation Load:** This algorithm is computationally light. It uses traditional computer version techniques that do not require annotations or training, which makes it suitable for processing large datasets with limited resources.
- **Improved Control:** By combining thresholding and contour-based masking, this algorithm provides consistent leaf segmentation results, even under variable lighting or background conditions. The same code can be used for any kind of rice leaf dataset, as it performs automated segmentation.

Figure 1 illustrates the workflow of this algorithm, which visually guides the process from grayscale conversion to green pixel isolation.

4.2 Generative Adverserial Network (GAN)

In this approach, we utilize Generative Adversarial Networks (GANs) to generate synthetic images based on two segmented datasets, D4 and D5. For each class in these datasets, we trained a separate GAN model to generate 500 synthetic images per class. The training process of GAN involves two primary components: the Generator and the Discriminator, which are trained in an adversarial manner.

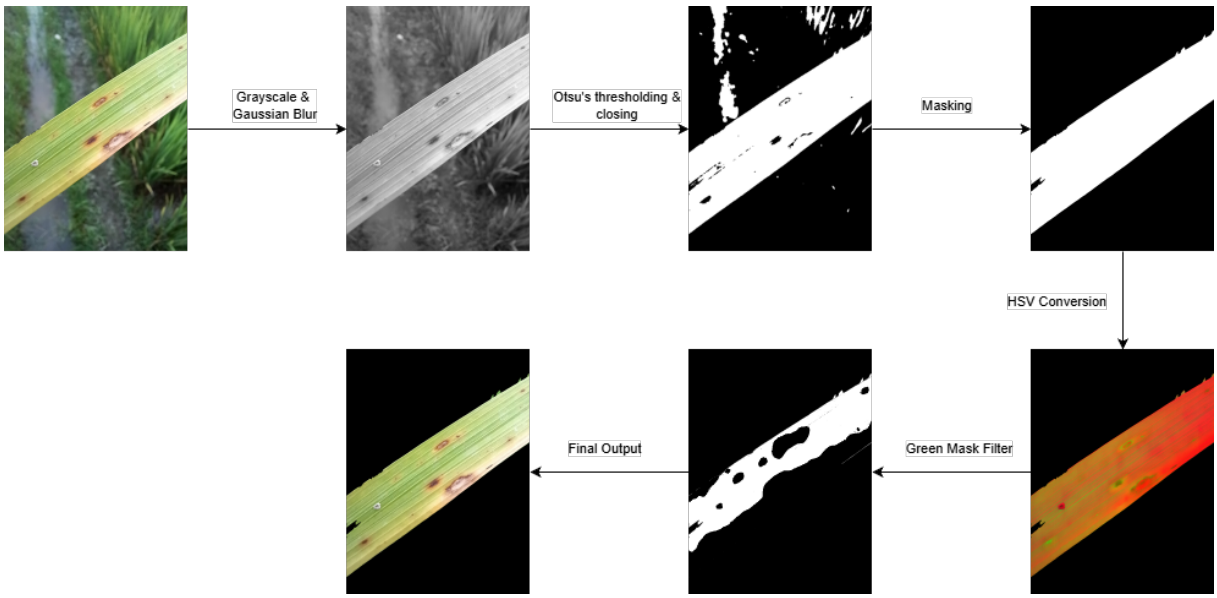

Figure 1: Process flow of Otsu Algorithm

Preprocessing with Otsu Segmentation The Otsu segmentation method was applied to both the D4 and D5 datasets to preprocess the images. This step was essential for isolating the regions of interest, such as leaf regions, which are crucial for disease detection. Segmentation was performed to ensure that the GAN focused on the relevant features of the image, removing background noise and irrelevant details. By segmenting the images first, we ensured that the GAN would generate synthetic images that more closely matched the characteristics of the real, leaf-focused images, which is key for improving model accuracy during training.

Training the Generator The Generator was trained on the segmented images to generate synthetic images for each class. During training, the Generator takes a latent vector $z \in \mathbb{R}^n$ sampled from a noise distribution (typically Gaussian) and generates an image $G(z)$. The Generator aims to minimize the Discriminator's ability to distinguish generated images from real ones. This process is framed as a minimax game, with the loss functions for both the Generator and Discriminator guiding the training. The objective for the Generator is to maximize the probability of the Discriminator classifying the generated images as real, which is mathematically expressed as:

$$\mathcal{L}_G = -\frac{1}{2}\mathbb{E}_{z\sim p_z(z)}[\log D(G(z))]$$

where $D(G(z))$ is the Discriminator's probability of the generated image being real. The Discriminator, in turn, is trained to minimize the error of classifying real versus fake images, as explained in the next paragraph.

Generating Synthetic Images and Model Training A total of 500 synthetic images were generated for each class of the datasets D4 and D5. These generated images were normalized to the range $[-1, 1]$ to match the input requirements of the models during training. The GAN-generated images were then used to augment the datasets, and several pre-trained models were trained on these synthetic images. For the D4 dataset, we used InceptionResNetV2 and InceptionDenseNet, while for the D5 dataset, models such as InceptionResNetV2, DenseNet201, MobileNetV2, ResNet152V2, and EfficientNetB3 were employed. These models were selected based on their performance in previous studies and served as a basis for evaluating the impact of GAN-generated data on model performance. The comparison between the models trained on real versus GAN-augmented datasets provided insights into how synthetic images affect classification accuracy and generalization.

Evaluation and Results After training the models on GAN-generated images, the models were evaluated on the original segmented D4 and D5 datasets to assess their generalization ability. The evaluation focused on comparing the performance of models trained on the original datasets ver-

sus those trained on the GAN-augmented datasets. The validation accuracy was calculated by determining the percentage of correctly classified images from the original dataset. This evaluation allowed us to quantify how the inclusion of synthetic data improved or affected the accuracy of the models. The results highlighted the potential of GANs in augmenting datasets to improve classification performance, particularly in scenarios where real data is limited.

4.3 YOLO + SAM-2

To address problems with the traditional segmentation approach, we present a pipeline that uses a combination of the well-renowned Yolov8 detector and the SOTA SAM-2 Segmentation model. Although Yolo could itself be used for detecting infected leaves, we figured out that just based on small spots, Yolo struggled in detecting the whole leaf particularly because it targeted the 95% green area of the leaf rather than the infected 5% (or even lower) part. As a result, the model predicted healthy leaves as infected too based on the green region. Therefore, rather than focusing on detecting the whole infected leaf, we trained Yolo specifically for detecting the spots on the leaves that held accountable for infection. However, while training, we figured out that the different infection classes of the leaves weren't only dependent on the spots that appeared but rather on the shape and color of that infected leaf too. Therefore, to utilize Yolo to its utmost while retaining the complete infected leaf, we applied a combination of Yolo and SAM-2. The former detects any spots that appear on a leaf and classifies it under only one category 'Anomaly'. The latter meanwhile segments any entity that it feels is a probable object and outputs a list of images each containing a single object. The bounding box output from the YOLO model is then applied on the original unsegmented image to extract regions of spots from the image. Next, that bounding box output is applied on each of these segmented images to extract regions from each of these images too. Then for each region, a similarity check is performed comparing the region extracted from the unsegmented image to the regions extracted from each of the segmented images. The segmented images having a higher similarity score are the leaves having the spots. This process creates a dataset that only has the segmented infected leaf along with a black background. This dataset is then used to train the seven different models for accuracy comparison. Using this approach as the base, we tried three different approaches built on top of the former in an incremental pipeline as an attempt to increase the classification accuracy.

First, we tried using the segmented leaves with black background only, however, as shown in table 6, the results weren't as expected. Next, we reduplicated the segmented leaves, this consisted of copying the leaves while preserving their shape and placing them in every available space of the image; wherever the complete space for the leaf to fit wasn't available, we placed the most part of the leaf that could fit in that space. We tried this using the intuition that instead of letting the model focus on one specific area only, that is training those weights only, spreading the leaves all over the image would train the weights throughout the image. For the weights that were still not trainable we applied background retention too to let the model learn from background cues. As shown in table 7, this showed significant increase in accuracy. The next section explains the proposed methodology in detail.

4.4 Reduplication with Background Retention

4.4.1 Methodology

The results from single leaf segmentation weren't as expected, rather, were the lowest on average among the three categories as shown in table 6. We figured out that this was due to the fact that for each example, only the weights where there is a positive pixel value are updated, which can further be explained by the hidden unit formulas below:

$$\hat{y} = \text{activation}(w \cdot x + b) \tag{1}$$

where activation() is a non-linear function (e.g., ReLU, sigmoid), w represents the weights, x the pixel values, and b the bias term.

By applying the chain rule, we get the gradient of the loss L with respect to w as:

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial \hat{y}} \cdot \text{activation}'(w \cdot x + b) \cdot x \tag{2}$$

where activation$'$() is the derivative of the activation function.

The weights are then updated as follows:

$$w := w - \eta \frac{\partial L}{\partial w} \tag{3}$$

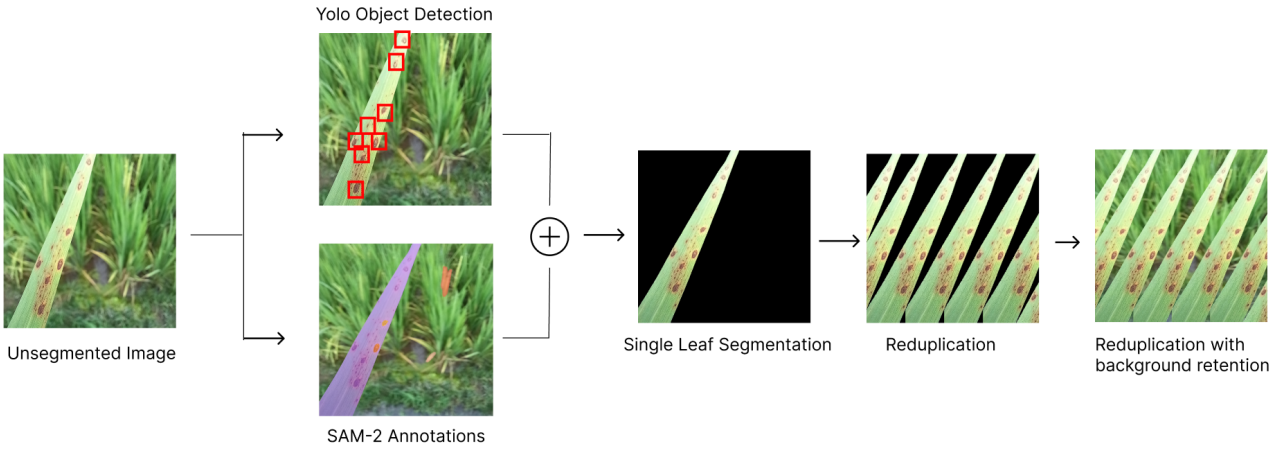

Figure 5: Visual Representation of the proposed approach

where η is the learning rate. If $x = 0$, then $\frac{\partial L}{\partial w} = 0$, meaning that weights corresponding to zero pixel values will not be updated. This explains that for each example, only 15% of image weights are trained. Consequently, if a test infected leaf may appear in any area of the image, the weights for that area would not have necessarily been generalized to all classes. This justifies the inferior accuracy compared to the other categories. To resolve this issue, we first used pixel replication, this copied the pixels from the leaf and placed them in the available spaces. This solved the issue for weights training however it distorted the distribution of spots and altered the shape of the copied objects. To avoid this, we then used object mapping, where instead of copying the pixels, we copied the whole object and placed it in available spaces. For areas, where the object would not fit in completely, we placed the maximum part of the object that could fit. This showed a remarkable improvement in accuracy. The only problem was that there were still regions where weight training was a problem. We then used background retention primarily for two reasons; firstly, to train the model weights throughout the image and secondly, to help the model gain more contextual understanding and focus on the objects of interest. This further improved the overall accuracy and hence set the benchmark for this classification task as shown in the table **??**. A visual representation of the pipeline is demonstrated in **??**.

4.4.2 Applicability

The issues mentioned above present several obstacles to making the approach deployable, however, if time is not a constraint and manual annotations can be done, the approach can be applied for automated disease detection in rice fields using drones. The annotations can be done on the data gathered from drone shots, because the goal is now to detect spots in rice fields, the dataset restriction would not be a problem. The SAM-2 segmentation can also be solved using the following approach: When comparing regions, a check can be placed on whether background is included in that segmented leaf or not; if it is, it means SAM-2 did not segment objects correctly, for this, the drone can repeatedly change the capturing angle slightly until that leaf is captured as an object. This pipeline ensures that the process of identifying infections in leaves becomes fully automated rather than manual analysis of leaves.

4.5 Classifiers

For our rice disease detection experiment, we made use of several transfer learning models, specifically InceptionResNetV2, DenseNet201, MobileNetV2, ResNet152V2, EfficientNetB3, InceptionDenseNet201, and VGG16. These models were used to classify disease types across different datasets. To investigate the model performance on the different preprocessing techniques, we trained each model on the original models, as well as the segmented datasets, and also on the GAN-generated datasets.

Model Loading and Preprocessing Each model was initialized using weights pre-traiend on the ImageNet dataset. This allowed it to make use of already learned features, which enabled faster converance and potentially higher accuracy. This technique, known as *transfer learning*, allows us to be able to make use of limited data, as

in the case of our rice disease datasets. It allows models to apply knowledge from vast datasets to the specific task at hand. The input shape for each model was adjusted between either $(150, 150, 3)$ pixels, or $(224, 224, 3)$ pixels, depending on the requirements of the chosen classifier. Each image was scaled to the same resolution for uniform input dimensions, and the input data was prepared using the TensorFlow `ImageDataGenerator` with a split of 80% training and 20% validation images. These sets were kept separate, as the validation set was used to determine the model's accuracy, and was not used in training.

Base Model Configuration In this step, the pre-trained model is loaded without its top (classification) layer. This allows us to customize the architecture for our rice disease classification task. This can be mathematically expressed as:

$$f_{base}(\mathbf{X}) = \text{BaseModel}(\mathbf{X})_{[:-1]}$$

where $\mathbf{X}$ represents the input image tensor. The final layers are replaced to enable classification of the number of disease categories according to the dataset being used, as this ranged from between 3 to 8 output classes based on the dataset.

Added Layers and Final Model Architecture To create the final classifier, we added a `GlobalAveragePooling2D` layer, which reduces spatial dimensions by averaging features along each channel. This operation is useful for collecting feature information across the entire image:

$$\mathbf{GAP}(\mathbf{X}) = \frac{1}{n} \sum_{i=1}^{n} x_i$$

where x_i are pixel values and n is the total number of pixels in the feature map.

Following the pooling layer, we added a dense layer with 1024 neurons and ReLU activation to introduce non-linearity. This allows for more complex patterns to be captured. The final output layer consists of 3-8 neurons with a softmax activation (the number depends on the output classes in the dataset). This output layer can be defined as:

$$y = \text{softmax}(W \cdot h + b)$$

where W and b are learned weights and biases, and h represents the features from the previous dense layer.

Compilation and Training The model was compiled using the Adam optimizer with a learning rate of 0.001. Adam optimizer assists training by adjusting learning rates dynamically, thus enhancing convergence rates. For multi-class classification, we employ the *categorical cross-entropy* loss function, which can be represented as:

$$\mathcal{L} = -\sum_{c=1}^{C} y_c \log(\hat{y}_c)$$

where y_c is the true label, $\hat{y}_c$ is the predicted probability for class c, and C is the number of classes.

After training, we evaluated the validation accuracy of the model to gauge it's generalization on unseen data. The best accuracy attained provided us with an assessment of the model's effectiveness across different preprocessing methods. This allows us to analyze and compare the performance on original, segmented, and GAN-generated datasets.

4.6 Experiments

4.6.1 Canny Edge Otsu Segmentation with GAN augmentation

In this section, we describe how the datasets were used to perform experimentation using the otsu segmentation approach.

Training and Validation The images for both the training and validation sets were segmented using the algorithm, as the algorithm is lightweight and can easily be used on any image. The process flow is as follows: the datasets are segmented, a model is trained on a dataset, a new image is entered, the new image is processed using the segmentation algorithm, the segmented image is passed to the trained model for detection. This approach simplifies the model's task of identifying the leaf area, ensuring that it only needs to focus on the important features. Moreover, by applying this approach on both training and validation, we allow the model to have consistent input during both training and deployment. Finally, for GAN augmentation, for each class in the **segmented** dataset, a GAN model was trained. The GAN models were then used to generate 500 images of each class. The transfer learning models were then trained on this new segmented and augmented dataset.

The following parameters were involved in the segmentation process:

- **Grayscale Conversion:** The OpenCV library was used to convert the image into grayscale. This simplifies the image data, and makes it easier to focus on the structure of the image.

- **Gaussian Blur:** The kernel size for the Gaussian Blur of (5, 5) was used as a balanced choice between removing noise and preserving essential image details. A larger kernel would overly smooth the image, which would result in potential loss of important features, while a smaller kernel might not effectively reduce noise.

- **Morphological Kernel Size:** The kernel for morphological closing was set to (4, 4). This was chosen to remove small holes or gaps in the contours without affecting the shape of the leaf too much. Larger kernel sizes might overly smooth the boundaries, while smaller sizes might leave small holes in the leaf mask.

- **Contour Area Check:** The algorithm identifies the largest contour in the image, assuming that this corresponds to the leaf. This decision is based on the assumption that the leaf will typically be the largest object in the image, which is valid for the dataset used.

- **Green Pixel Threshold:** The threshold for the green pixel ratio was set to 1%. This step is designed to ensure that no images that fail the segmentation process are able to get past into the produced dataset. It ensures that there are no images that do not contain a leaf at all. If the resultant image does not contain 1% green pixels, it excludes the image from the processed dataset.

Model Training The segmented dataset was then used to train a model for disease detection. It used the same parameters for the non-segmented approach, as well as the same number of output classes. This is so that the difference in accuracy can be fairly compared. After this, the augmented datasets generated by the GAN models were trained on the same models using the same parameters, for fair comparison.

5 Results

In this section, we will discuss the accuracy results of models trained on the original datasets, as well as the accuracy results of models trained on segmented and augmented datasets.

Table 2 shows the results from the experiments on the dataset D1. We can see that due to the large number of images in the dataset, the accuracy achieved by transfer learning models was already remarkable. Therefore, segmentation and augmentation were not effective here as there is no data refinement required.

Table 3 represents the results from experiments performed on dataset D2, which was also a dataset with an excessively large number of images. Thus, data segmentation and augmentation were not effective here either as the accuracy achieved by the original datasets were already considerably high.

Table 4 displays the results for the experiments performed on the dataset D3. We can see significant improvements in accuracy using the Canny Edge Otsu segmentation for 4 of the 5 models used on the dataset. These increases in accuracy go up to as high as a 24% increase in accuracy of the model for the InceptionResNetV2 model. Furthermore, we can see increases in accuracy using the YOLO + SAM-2 segmentation approach on 3 of the 5 models used. Although these increases are not as dramatic as the ones produced by the other approach, the results show the positive effect of segmentation on the model's accuracy.

Table 5 shows results for the experiments performed on the Dataset D4. We can see an increase in accuracy in 3 of the 5 models, achieved using the Canny Edge Otsu segmentation approach. The increases in accuracy go up as much as 11% for the ResNet152V2 model. The InceptionResNetV2 and Densnet201 models also achieve a significant increase in model accuracy using segmentation.

Table 6 shows results for the experiments performed on dataset D5. In this dataset, we can see an increase in accuracy using the Canny Edge Otsu segmentation on 4 of the 7 models that we used. The highest increase in accuracy can be seen in the VGG16 model, an increase of 31.6%. The other 3 models that see an increase include the MobileNetV2, ResNet152V2, and InceptionDensnet201. The YOLO + SAM-2 segmentation approach itself is not effective here as it is only able to increase the accuracy on the VGG16 model, however, we will see in the next table how the

Table 2: Accuracy Results on Dataset D1

Model Used	**Original Accuracy**	**Accuracy using Canny Edge Otsu segmentation**
InceptionResNetV2	99%	98%
Densnet201	99%	96%
MobileNetV2	77%	81%
ResNet152V2	98%	93%
EfficientNetB3	100%	100%

Table 3: Accuracy Results on Dataset D2

Model Used	**Original Accuracy**	**Accuracy using Canny Edge Otsu segmentation**
InceptionResNetV2	99%	100%
Densnet201	99%	99%
MobileNetV2	90%	69%
ResNet152V2	98%	94%
EfficientNetB3	100%	100%

Table 4: Accuracy Results on Dataset D3

Model Used	**Original Accuracy**	**Accuracy using Canny Edge Otsu segmentation**	**Accuracy using YOLO + SAM-2 segmentation**
InceptionResNetV2	34%	58%	38%
Densnet201	36%	47%	35%
MobileNetV2	28%	38%	36%
ResNet152V2	23%	36%	43%
EfficientNetB3	58%	47%	39%

Table 5: Accuracy Results on Dataset D3

Model Used	**Original Accuracy**	**Accuracy using Canny Edge Otsu segmentation**
InceptionResNetV2	60%	65%
Densnet201	64%	68%
MobileNetV2	29%	26%
ResNet152V2	46%	57%
EfficientNetB3	66%	61%

Table 6: Accuracy Results on Dataset D5

Model Used	**Original Accuracy**	**Accuracy using Canny Edge Otsu segmentation**	**Accuracy using YOLO + SAM-2 segmentation**
InceptionResNetV2	74%	73.2%	63%
Densnet201	73.6%	69.4%	70%
MobileNetV2	38.5%	41.4%	44%
ResNet152V2	51.4%	62.7%	50.1%
EfficientNetB3	79.1%	76.6%	64%
InceptionDensnet201	65.7%	73%	62%
VGG16	27%	58.6%	41%

Reduplication with Background Retention augmentation technique changes the results dramatically.

Table 7 shows the results for the experiments performed on dataset D5 with the addition of reduplication augmentation technique.

Lastly, table 8 shows the results for the experiments performed on dataset D4 and dataset D5 with the addition of GAN. For dataset D4, we can see extraordinary improvements in accuracy, with

Table 7: Accuracy Results using Reduplication with Background Retention on D5

Model Used	Original Accuracy	Accuracy using Canny Edge Otsu segmentation	Accuracy with YOLO + SAM-2 segmentation	YOLO + SAM-2 (Reduplication + Background retention)
InceptionResNetV2	74.2%	72.4%	**75%**	72.5%
Densnet201	72.4%	65.4%	72%	**73.5%**
MobileNetV2	52.7%	53.3%	63%	**66.9%**
ResNet152V2	72.4%	59.2%	64%	**75.7%**
EfficientNetB3	77.4%	73.0%	75%	**78.1%**
VGG16	**78.1%**	72.2%	73.5%	70.1%
InceptionDenseNet201	**79.1%**	73.1%	75.5%	77.8%
Yolov8n-cls	75.8%	71.9%	77.7%	**81.1%**

Table 8: Accuracy Results using GAN

Dataset Name	Model Used	Original Accuracy	Accuracy using Canny Edge Otsu segmentation	Accuracy using Canny Edge Otsu segmentation with GAN augmentation
D4	InceptionResNetV2	60%	65%	93.2%
D4	InceptionDenseNet	56%	53%	91%
D5	MobileNetV2	29%	27%	49.8%
D5	ResNet152V2	47%	35%	54.5%

a 33.2% increase in accuracy for the InceptionResNetV2 model over the original accuracy. We also see an impressive 35% increase in accuracy using the InceptionDenseNet model. For dataset D5, we see a remarkable 20.8% increase for the MobileNetV2 model, and a 7.5% increase in accuracy for the ResNet152V2 model over the original accuracies. These results were produced by: segmenting the dataset using the Canny Edge Otsu segmentation algorithm, generating synthetic data from the segmented datasets, and finally training the models on this synthetic data. The accuracy was then measured by model inference on the original data. All four of these results are remarkably higher than the accuracy achieved by the Canny Edge Otsu segmentation alone.

6 Discussion

The accuracy improvements achieved through different segmentation and augmentation techniques highlight the role of targeted image processing in enhancing model performance. Each approach contributes to the improvement in model accuracy in different ways. This section discusses the effects of the distinct approaches, the comparison between the two approaches, and future work.

6.1 Effect of Segmentation and Augmentation

The application of the segmentation and augmentation techniques led to notable accuracy increases across a variety of different datasets and models. The segmentation techniques (Canny Edge Otsu and YOLO + SAM-2) isolates disease-affected areas, which allowed the models to focus on essential patterns in the images with reduced distractions from background noise. Augmentation methods on the other hand, particularly GAN-based and reduplication, significantly expanded the dataset diversity and volume. By enabling the models to learn from a wider range of disease patterns, these two methods improved performance by a dramatic amount.

The approaches had the following effects on the dataset, which led to accuracy improvements:

- **Noise Reduction:** By isolating the diseased leaf, segmentation was able to minimize background interference, which made it easier for models to learn relevant disease features. This reduction in noise allowed for a more refined learning process, as shown by the Canny Edge Otsu segmentation's success in datasets like D3 and D4.
- **Increased Data Volume** The use of GANs to generate synthetic images expanded the dataset's volume, which improved model generalization by an extraordinary amount, especially in the case of dataset D4. This increase in data, powered by a segmented dataset, allowed models to improve on their learning process which led to a substantial accuracy boost.
- **Enhanced Data Diversity:** The reduplication method created more instances of disease patterns in each image, which helped reinforce pattern recognition for the models. This approach proved effective on D5, where repeated exposure to similar patterns improved accuracy across various models.

Overall, segmentation enhanced the focus on disease areas, while augmentation amplified data variety, both of which played essential roles in the accuracy improvements.

6.2 Comparison of Segmentation Techniques

The segmentation techniques each employed unique strengths:

- **Canny Edge Otsu Segmentation:** This technique's strength lies in it's ability to automatically determine the target leaf, and apply parameters to extract the diseased leaf based on any background conditions. It is a fully automated process with an extremely low computation power requirement. It was especially effective on small-sized datasets, as it shows the best results when used to improve data that is scarce. It's noise-reduction capability helped the models focus exclusively on the leaf's disease features, which improved training outcomes. However, it has a drawback that it fails to execute if there is no leaf in the target of the image, or if the leaf is too small as compared to the background.

- **YOLO + SAM-2 Segmentation:** The YOLO + SAM-2 approach proved beneficial as it is able to identify and extract diseased leaves at a much higher precision as compared to the Canny Edge Otsu approach, especially when background elements are complex. Although the accuracy gains were smaller when used alone, this approach was highly effective when combined with reduplication, where it was able to produce enhanced results by increasing the model's exposure to the diseased leaves. However, it has a drawback that it requires much more computation resources and annotations on the dataset to train the YOLO model.

6.3 Comparison of Augmentation Techniques

The two augmentation methods are compared below:

- **GAN-based Augmentation:** By generating synthetic images of segmented diseased leaves, the GAN approach was able to multiply the volume of the dataset using refined images, which had gone through the segmentation process. This technique resulted in extraordinary accuracy improvements, as seenin datasets D4 and D5, where a model achieved an up to 35% gain in accuracy. This augmentation approach allowed the trained model to be much more powerful, and was especially effective when the dataset size was extremely small. However, this approach requires extensive computation resources to train the GAN models for accurate synthetic data generation.

- **Reduplication with Background Retention:** This technique involved using the segmented leaf and augmenting the image to contain multiple instances of the segmented leaf without any background. This technique was especially effective when used with the YOLO + SAM-2 segmentation technique as that approach ensured that the output image would contain only the segmented leaf. This resulted in impressive gains in accuracy across various models in dataset D5, where models benefited from multiplied exposure to the target area.

6.3.1 Future Work

The impressive accuracy improvements achieved through segmenation and augmentation indicate promising directions for future research, especially in the case of augmentation. The dramatic results achieved through the combination of the segmentation and augmentation techniques suggest that exploring these augmentation techniques to improve model accuracy, especially GAN-based synthetic data generation and reduplication, could produce even better outcomes. Future work can focus on developing hybrid approaches that integrate both these techniques more deeply, potentially leveraging other augmentation techniques or generative models to refine disease feature isolation. This would result in datasets being modified to be much more feature-rich, which could produce better results than if the original datasets were used to train models.

7 Conclusion

In conclusion, this study presents an effective approach for improving rice disease detection by combining segmentation and augmentation techniques using transfer learning models. Since rice disease data is scarce, these techniques can be utilized to achieve dramatic improvements in model-based detection performance. This paper presents

two segmentation pipelines to isolate disease-specific areas on rice leaves, and two augmentation techniques used to increase dataset volume and quality. All of these approaches achieved significant enhancements to model accuracy and performance. Through experimentation, we demonstrated that these enhancements allowed models like DenseNet201, EfficientNetB3, and Inception-ResNetV2 to achieve high accuracy on smaller datasets. The results highlight the potential of isolating target areas and expanding dataset diversity in scenarios with limited agricultural data, as this approach could be translated to other plant-based disease detection systems.

References

[1] J. Kihoro, J. Mukundi, H. Murage, E. Ateka, and D. Makihara, "Investigating the impact of rice blast disease on the livelihood of the local farmers in greater mwea region of kenya," *SpringerPlus*, vol. 2, p. 308, 12 2013.

[2] B. Gülmez, "Advancements in rice disease detection through convolutional neural networks: A comprehensive review," *Heliyon*, vol. 10, no. 12, p. e33328, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405844024093599

[3] P. Kulkarni and S. Shastri, "Rice leaf diseases detection using machine learning," *Journal of Scientific Research and Technology*, pp. 17–22, 01 2024.

[4] H. Liu, Y. Cui, J. Wang, and H. Yu, "Analysis and research on rice disease identification method based on deep learning," *Sustainability*, vol. 15, no. 12, 2023. [Online]. Available: https://www.mdpi.com/2071-1050/15/12/9321

[5] Rukhsar and S. K. Upadhyay, "Rice leaves disease detection and classification using transfer learning technique," in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2022, pp. 2151–2156.

[6] T. Yang, S. Zhou, A. Xu, J. Ye, and J. Yin, "An approach for plant leaf image segmentation based on yolov8 and the improved deeplabv3+," *Plants*, vol. 12, no. 19, 2023. [Online]. Available: https://www.mdpi.com/2223-7747/12/19/3438

[7] V. Singh and A. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques," *Information Processing in Agriculture*, vol. 4, no. 1, pp. 41–49, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214317316300154

[8] M. Shoaib, T. Hussain, B. Shah, I. Ullah, S. M. Shah, F. Ali, and S. H. Park, "Deep learning-based segmentation and classification of leaf images for detection of tomato plant disease," *Frontiers in Plant Science*, vol. 13, 2022. [Online]. Available: https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2022.1031748

[9] S. Zhang and C. Zhang, "Modified u-net for plant diseased leaf image segmentation," *Computers and Electronics in Agriculture*, vol. 204, p. 107511, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169922008195

[10] H. Yang, X. Deng, H. Shen, Q. Lei, S. Zhang, and N. Liu, "Disease detection and identification of rice leaf based on improved detection transformer," *Agriculture*, vol. 13, no. 7, 2023. [Online]. Available: https://www.mdpi.com/2077-0472/13/7/1361

[11] K. Dutta, D. Talukdar, and S. S. Bora, "Segmentation of unhealthy leaves in cruciferous crops for early disease detection using vegetative indices and otsu thresholding of aerial images," *Measurement*, vol. 189, p. 110478, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224121013622

[12] P. Ramesh Babu, A. Srikrishna, and V. Gera, "Diagnosis of tomato leaf disease using otsu multi-threshold image segmentation-based chimp optimization algorithm and lenet-5 classifier," *Journal of Plant Diseases and Protection*, vol. 131, 06 2024.

[13] A. Ks and A. Sahayadhas, "Automatic rice leaf disease segmentation using image processing techniques," *International Journal of Engineering and Technology(UAE)*, vol. 7, pp. 182–185, 08 2018.

[14] J. Barbedo, "A new automatic method for disease symptom segmentation in digital photographs of plant leaves," *European Journal of Plant Pathology*, vol. 147, pp. 349–364, 02 2017.