A project report on

ANALYSIS OF POTATO LEAF DISEASE DATASET IN AN UNCONTROLLED ENVIRONMENT UTILIZING CUSTOMISED YOLOv8 CLASSIFIERS

Submitted in partial fulfillment for the award of the degree of

Bachelor of Technology in Computer Science and Engineering with Specialization in Cyber Physical Systems

by

PREYASH (20BPS1022)

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2024

ANALYSIS OF POTATO LEAF DISEASE DATASET IN AN UNCONTROLLED ENVIRONMENT UTILIZING CUSTOMISED YOLOv8 CLASSIFIERS

Submitted in partial fulfillment for the award of the degree of

Bachelor of Technology in Computer Science and Engineering with Specialization in Cyber Physical Systems

by

PREYASH (20BPS1022)

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)
CHENNAI

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

April, 2024

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

DECLARATION

I hereby declare that the thesis entitled "ANALYSIS OF POTATO LEAF DISEASE DATASET IN AN UNCONTROLLED ENVIRONMENT UTILIZING CUSTOMISED YOLOv8 CLASSIFIERS" submitted by me, for the award of the degree of Bachelor of Technology in Computer Science and Engineering with Specialization in Cyber Physical Systems, Vellore Institute of Technology, Chennai is a record of bonafide work carried out by me under the supervision of Dr Suganya G.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Chennai

Date: 26/04/24

Preyash

Signature of the Candidate

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CHENNAI

School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled **"Analysis of Potato leaf disease dataset in an uncontrolled environment utilizing customised YOLOv8 Classifiers"** is prepared and submitted by **Preyash (20BPS1022)** to Vellore Institute of Technology, Chennai, in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering with Specialization in Cyber Physical Systems** programme is a bonafide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma and the same is certified.

Signature of the Guide: [illegible]

Name: Dr. Suganya G

Date: 23/4/24 .

[illegible]	[illegible]
Signature of the Internal Examiner	Signature of the External Examiner
Name: SUNDARAM	Name: Surendra Reddy K
Date: 26/4/24	Date: 26/04/2024

Approved by the Head of Department,
B.Tech. CSE with Specialization in Cyber Physical Systems

[illegible]

Name: Dr. Renuka Devi S

Date: 23/04/24

(Seal of SCOPE)

VELLORE INSTITUTE OF TECHNOLOGY
SCHOOL OF COMPUTER SCIENCE & ENGINEERING

ABSTRACT

Potatoes are essential for global food security, and effective management of potato diseases is crucial. This study used a novel uncontrolled environment potato leaf disease dataset to develop a custom YOLOv8 model for potato disease detection. We employed various preprocessing techniques like Histogram Equalization (HE), Contrast Limited Adaptive Histogram Equalization (CLAHE), Hue Saturation Value (HSV) conversion, Median denoising, and Gaussian denoising to enhance image quality and feature extraction. The base paper achieved a maximum accuracy of 73% using EfficientNetV2B3, while our proposed model, YOLOv8_STN, achieved an outstanding accuracy of 91% with the extra-large variant of YOLOv8_STN preprocessed using HSV. As the dataset had class imbalance, we relied on the F1 score as a more reliable evaluation metric, where our model achieved an F1 score of 90%, surpassing the base paper's highest F1 score of 73%.

Furthermore, we performed a comparative analysis with CNNs like ResNet, DenseNet, and VGG to evaluate our model's generalisation abilities and efficacy in detecting potato leaf disease. We also evaluated the Plant Village dataset, confirming our model's superior performance. These findings underscore the potential of deep learning techniques, especially YOLOv8_STN, to revolutionise agricultural disease management and improve crop yield and food security worldwide.

ACKNOWLEDGEMENT

It is my pleasure to express with deep sense of gratitude to Dr. Suganya G, Associate Professor Senior, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, for her constant guidance, continual encouragement, understanding; more than all, she taught me patience in my endeavour. My association with her is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Computer Science and Machine Learning.

It is with gratitude that I would like to extend my thanks to the visionary leader Dr G. Viswanathan our Honorable Chancellor, Mr Sankar Viswanathan, Dr Sekar Viswanathan, Dr G V Selvam, Vice President Dr Sandhya Pentareddy, Executive Director, Ms Kadhambari S. Viswanathan, Assistant Vice-President, Dr V. S. Kanchana Bhaaskaran Vice-Chancellor, Dr T. Thyagarajan Pro-Vice Chancellor, VIT Chennai and Dr P. K. Manoharan, Additional Registrar for providing an exceptional working environment and inspiring all of us during the tenure of the course.

Special mention to Dr. Ganesan R, Dean, Dr. Parvathi R, Associate Dean Academics, Dr. Geetha S, Associate Dean Research, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai for spending their valuable time and efforts in sharing their knowledge and for helping us in every aspect.

In jubilant state, I express ingeniously my whole-hearted thanks to Dr. Renuka Devi S, Head of the Department, B.Tech. CSE with Specialization in Cyber Physical Systems and the Project Coordinators for their valuable support and encouragement to take up and complete the thesis.

My sincere thanks to all the faculties and staffs at Vellore Institute of Technology, Chennai who helped me acquire the requisite knowledge. I would like to thank my parents for their support. It is indeed a pleasure to thank my friends who encouraged me to take up and complete this task.

Place: Chennai

Date: 26-04-2024 **Preyash**

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

CLAHE	-	Contrast Limited Adaptive Histogram Equalization
HSV	-	Hue, Saturation, Value
HE	-	Histogram Equalization
YOLO	-	You Look Only Once
CNN	-	Convolutional Neural Network
STN	-	Spatial Transformer Network
TP	-	True Positive
TN	-	True Negative
FP	-	False Positive
FN	-	False Negative

Chapter 1

Introduction

1.1 OVERVIEW

Potatoes stand out as one of the most extensively consumed vegetables worldwide, with their significance deeply ingrained in diverse cultures, underscoring the critical nature of potato production. According to statistics from the Food and Agriculture Organization of the United Nations (FAO), global potato production reached approximately 375 million tons in 2022, highlighting its substantial contribution to food security and agricultural economies [1]. However, potatoes are susceptible to diseases like many crops, posing significant challenges to production yields. Research endeavours in this domain seek to leverage computer vision technologies to accurately predict and diagnose potato plant diseases. This research aims to facilitate timely interventions and enable farmers to implement effective measures to safeguard their crops. By harnessing the power of machine learning algorithms and image processing techniques, such as convolutional neural networks (CNNs) and YOLO (You Look Only Once), researchers are paving the way for a more efficient and proactive approach to disease management in potato cultivation. This technological advancement holds promise in enhancing agricultural sustainability, reducing crop losses, and ensuring food security globally in potato-dependent regions.

1.2 POTATO AND SUSTAINABILITY GOAL 13

Potatoes contribute positively to Sustainable Development Goal 13 (SDG 13) by aligning with efforts to reduce carbon footprints in agricultural production. Potatoes support SDG 13 through their relatively low carbon footprint during production compared to other staple crops. According to studies and reports, potatoes have a lower carbon footprint per kilogram compared to crops like rice, wheat, or maise. This is attributed to several factors, such as efficient water use, less land requirement, and lower energy inputs during cultivation and processing.

Potatoes are known for their ability to grow in diverse climates and soil types, reducing the need for extensive irrigation and land clearing. Additionally, they have a shorter growing cycle compared to many other crops, leading to reduced energy consumption during cultivation and harvesting. Furthermore, potatoes require fewer inputs, such as fertilisers and pesticides, than other crops, contributing to lower greenhouse gas emissions associated with agricultural activities.

By promoting the cultivation and consumption of potatoes with a smaller environmental footprint, agricultural systems can become more sustainable and aligned with SDG 13's targets. This includes reducing greenhouse gas emissions, enhancing climate resilience, promoting sustainable land use, and adopting eco-friendly farming practices. Therefore, highlighting potatoes' benefits in terms of lower carbon footprints can be a compelling argument for their inclusion in sustainable agriculture strategies aimed at achieving SDG 13.

1.3 COMPUTER VISION IN PLANT LEAF DISEASE IMAGE ANALYSIS

Computer vision plays a crucial role in plant leaf disease image analysis, revolutionising the way agricultural experts detect, diagnose, and manage plant diseases. This innovative approach uses algorithms and machine learning techniques to analyse digital images of plant leaves, identifying symptoms of diseases such as fungal infections, nutrient deficiencies, and pest damage. One critical method used in plant leaf disease image analysis is Convolutional Neural Networks (CNNs), which excel in image recognition tasks by automatically learning relevant features from the input data. CNNs are trained on large datasets of labelled plant images, enabling them to accurately classify and detect diseases based on visual patterns and characteristics.

The importance of computer vision in plant leaf disease analysis cannot be overstated. It provides a rapid and non-destructive method for early disease detection, allowing farmers and agronomists to take timely action to mitigate crop losses and prevent disease spread.

By automating the detection process, computer vision systems enable efficient monitoring of large agricultural fields, reducing the reliance on manual inspection and improving overall crop management practices.

Computer vision has proven highly effective in analysing plant diseases, thanks to its ability to accurately and reliably identify diseases. This is a significant improvement over the traditional visual inspection method. Additionally, deep learning models like CNNs have significantly improved the performance and scalability of computer vision solutions, making it possible to deploy them in real-world agricultural settings.

Advanced techniques like You Only Look Once (YOLO) algorithms have been applied to plant leaf disease image analysis and CNNs. These algorithms are known for their speed and accuracy in object detection tasks, making it possible to process large image datasets in real-time. This feature is precious for continuously monitoring crops and intervening early against rapidly spreading diseases.

Computer vision-based image analysis transforms modern agriculture, using cutting-edge tech to enhance disease management, improve crop productivity, and support sustainable farming practices.

1.3.1 CNN FOR PLANT LEAF DISEASE ANALYSIS

Convolutional Neural Networks (CNNs) have revolutionised plant leaf disease analysis, making them a powerful tool in agriculture. Plant diseases can be automatically identified and classified based on visual cues from leaf images using deep learning techniques with CNNs. Their hierarchical architecture allows them to learn intricate patterns and features crucial for disease diagnosis, reducing reliance on manual inspection and providing rapid, accurate assessments. CNNs have proven particularly effective in handling large datasets of diverse plant species and diseases, showcasing their adaptability and robustness in real-world agricultural scenarios. Their deployment in plant leaf disease analysis holds immense promise for enhancing crop management, enabling early disease detection, optimising treatment strategies, and ultimately contributing to improved agricultural productivity and sustainability.

1.3.2 YOLO FOR PLANT LEAF DISEASE ANALYSIS

YOLO, which stands for "You Only Look Once," is a state-of-the-art object detection algorithm in computer vision and deep learning. Unlike traditional object detection methods that involve multiple passes through an image to detect objects, YOLO processes the entire image in one pass, making it incredibly fast and efficient. YOLO offers several advantages for plant leaf disease analysis:

1. Speed: YOLO is known for its speed and real-time processing capabilities. This is crucial in agricultural applications where rapid detection of plant diseases is essential to prevent their spread and minimise crop damage.

2. Accuracy: Despite its speed, YOLO maintains high accuracy in detecting objects, including disease symptoms on plant leaves. Its deep learning architecture enables it to learn and recognise complex patterns and features associated with various plant diseases.

3. Efficiency: YOLO's efficiency lies in its ability to detect multiple objects simultaneously within an image. This is particularly beneficial in agricultural settings where leaves may exhibit various symptoms or diseases, allowing for comprehensive analysis in a single pass.

4. Scalability: YOLO is scalable and can efficiently handle large datasets of leaf images. This scalability is crucial for analysing vast agricultural fields and monitoring numerous plants for disease symptoms.

5. Automation: By automating the process of disease symptom detection on plant leaves, YOLO reduces the need for manual inspection and human intervention. This automation saves time and resources, enabling early intervention and targeted treatment strategies.

YOLO algorithms are renowned for their speed and accuracy in object detection tasks, enabling real-time processing of large-scale leaf image datasets. This efficiency is particularly advantageous in continuous monitoring and early detection of plant diseases, allowing farmers and agronomists to identify and address issues before they escalate promptly. YOLO algorithms leverage profound learning principles to efficiently detect disease symptoms and anomalies in leaf images, providing valuable insights into disease prevalence, severity, and crop distribution. Integrating YOLO algorithms in plant leaf disease analysis represents a significant step towards precision agriculture, optimising resource allocation, improving crop health, and fostering sustainable farming practices.

1.4 LITERATURE SURVEY

Over the years, remarkable progress has been made in plant leaf disease classification. This can be credited to the rise in leaf image availability and the growing interest in using computer vision to improve analysis. The succeeding section will summarise the potato and other plant leaf disease identification studies using Machine Learning (ML) and Deep Learning (DL) methods.

Wang et al. introduced UAV-YOLOv8, a novel small-object-detection model tailored for UAV aerial photography scenarios, which was published in Sensors [3]. The paper addresses challenges faced by existing object detection models embedded in UAVs, particularly concerning the high proportion of small objects in UAV images and limited platform resources leading to low accuracy. UAV-YOLOv8 optimises the YOLOv8 model through several vital enhancements to tackle these issues. Firstly, it employs the Wise-IoU (WIoU) v3 bounding box regression loss and a wise gradient allocation strategy to focus on common-quality samples, thus enhancing the model's localisation ability. Secondly, an attention mechanism called BiFormer is introduced to optimise the backbone network, improving attention to critical information. Lastly, a feature processing module named Focal FasterNet block (FFNB) and two new detection scales are proposed, facilitating multiscale feature fusion and enhancing detection performance, particularly for small objects. Experimental results demonstrate that UAV-YOLOv8 outperforms baseline models with a mean detection accuracy 7.7% higher than the baseline, showcasing its effectiveness in small-object detection within UAV aerial photography scenarios.

Wu and Dong introduced YOLO-SE, an innovative approach based on the YOLOv8 network, designed to address remote sensing object detection and recognition challenges [4]. Their "YOLO-SE: Improved YOLOv8 for Remote Sensing Object Detection and Recognition" research was published in Applied Sciences. YOLO-SE incorporates key advancements to enhance object detection performance, including introducing a lightweight convolution SEConv, the Efficient Multi-Scale Attention (EMA) mechanism, and the Wise-IoU bounding box loss function. These innovations collectively improve the network's ability to detect small-sized targets and objects across diverse scales. The study demonstrated significant performance gains, with YOLO-SE achieving an average precision at IoU threshold 0.5 (AP50) of 86.5% on the SIMD dataset, outperforming YOLOv8 by 2.1% and surpassing the state-of-the-art model by 0.91%. Additionally, experiments on the NWPU VHR-10 dataset validated YOLO-SE's superiority with an accuracy of 94.9%, surpassing YOLOv8 by 2.6%. These results highlight YOLO-SE as a promising solution in deep learning-based remote sensing object detection applications.

Sary et al. conducted a study titled "Performance Comparison of YOLOv5 and YOLOv8 Architectures in Human Detection using Aerial Images," published in Ultima Computing: Jurnal Sistem Komputer [5]. Their research compared the performance metrics of YOLOv5 and YOLOv8, explicitly focusing on precision, recall, and F1-score values. The study's findings revealed that YOLOv8 demonstrated superior precision and F1-score results compared to YOLOv5, although YOLOv5 exhibited higher recall rates. This highlights the trade-off between accuracy and recall in object detection models and provides valuable insights for optimising such frameworks in aerial surveillance and emergency response applications.

Al-Adhaileh et al. presented a fine-tuned CNN architecture for accurately detecting potato blight, a destructive global disease affecting potato crops [6]. The customised CNN model was designed to enhance accuracy while minimising trainable parameters, computation time, and information loss. Their approach involved data balancing, augmentation, and pre-processing steps, focusing on reducing information loss typically associated with pooling layers in CNN architectures. The proposed model outperformed other machine and deep learning algorithms, achieving an impressive overall accuracy of 99% with minimal trainable parameters and a short training time of 183 seconds.

Arshad et al. introduced PLDPNet, a novel hybrid deep learning framework designed for accurate prediction of potato leaf diseases, as detailed in their publication in Alexandria Engineering Journal [7]. The framework encompasses various stages: image collection, pre-processing, segmentation, feature extraction and fusion, and classification. Notably, PLDPNet leverages an ensemble approach by combining features from established models like VGG19 and Inception-V3, followed by a transformation using Vision Transformer (ViT) for final prediction. The ViT is chosen for its self-attention mechanism, which enables precise object detection and reduced reliance on vision-specific biases. The proposed framework achieved impressive results with an overall accuracy of 98.66% and an F1-score of 96.33% on a public potato leaf dataset. Additionally, validation studies on Apple and tomato datasets demonstrated accuracies of 96.42% and 94.25%, respectively, affirming the effectiveness and versatility of the PLDPNet framework in agricultural disease prediction.

Jung and Choi conducted research titled "Improved YOLOv5: Efficient Object Detection Using Drone Images under Various Conditions," published in Applied Sciences [8]. Their study aimed to enhance object detection performance, particularly in challenging environments such as those encountered during illegal immigration monitoring, industrial and natural disasters, and search operations for missing individuals or objects. They utilised the F11 4K PRO drone and the VisDrone dataset to capture experimental data under varying conditions, including changes in drone altitude, low-light scenarios, and diverse environmental settings. The researchers proposed an improved YOLOv5 model, YOLOv5_Ours, and evaluated its performance metrics against the original YOLOv5 model. Key indicators such as precision, recall, F-1 score, and mAP (0.5) were used for comparison, with YOLOv5_Ours demonstrating enhanced mAP (0.5) and function loss values compared to the original YOLOv5 model. This study contributes valuable insights into optimising object detection algorithms for efficient drone-based surveillance and response operations.

Rashid et al. presented a Multi-Level Deep Learning Model for Potato Leaf Disease Recognition, addressing challenges in early-stage detection due to variations in crop species, disease symptoms, and environmental factors [9]. The model incorporates a two-level approach: firstly, utilising YOLOv5 for potato leaf extraction from plant images, and secondly, employing a novel convolutional neural network for early and late blight disease detection.

Sudi et al. presented "A CNN Model for Disease Detection in Potato Leaves" [10]. Focused on addressing concerns about crop production rates in India, the study proposes a strategy to enhance agricultural yields and prevent plant disease infections. Utilising deep neural networks, specifically Convolutional Neural Networks (CNN), the study explores various experiments, emphasising CNN's superior performance in identifying leaf diseases compared to other methods. The CNN model achieves high accuracy in disease detection.

Iqbal et al. presented "Detection of Potato Disease Using Image Segmentation and Machine Learning" [11]. The study focuses on Potato Leaf diseases, particularly Early Blight (EB) and Late Blight (LB), which challenge potato plant growth. The paper proposes an automatic system combining image processing and machine learning to identify and classify potato leaf diseases. Image segmentation is performed on 450 images from the Plant Village database, and seven classifier algorithms are employed for recognition and classification. The Random Forest classifier achieves an impressive 97% accuracy, showcasing the efficacy of the proposed approach in automatic plant leaf disease detection.

Agarwala et al. proposed a deep learning-based approach titled "ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network" [12]. The authors applied a Convolutional Neural Network (CNN) with three convolution and three max-pooling layers, followed by two fully connected layers, achieving notable disease detection and classification results. Their experimental outcomes demonstrated the efficacy of the proposed model, surpassing pre-trained models such as VGG16, InceptionV3, and MobileNet. The classification accuracy ranged from 76% to 100% across different classes, with an average accuracy of 91.2% for identifying nine diseases and one healthy class.

Shah et al. introduced ResTS, an interpretable architecture for plant disease detection based on Convolutional Neural Networks. ResTS utilises two classifiers (ResTeacher and ResStudent) and a decoder, training them reciprocally for enhanced interpretability [13]. After each convolution operation, the architecture's residual connections and batch normalisation preserve gradients, preventing issues like vanishing or exploding gradients. ResTS outperformed the Teacher/Student architecture (F1 score: 0.991 vs. 0.972), providing finer visualisations of disease symptoms. The approach, tested on the PlantVillage dataset with 54,306 images, showcases swift convergence and increased reliability.

Tambe et al. proposed a Convolutional Neural Network (CNN) approach for classifying potato leaf diseases using Deep Learning [14]. Their study involved preprocessing leaf image data, training a CNN model, and evaluating its accuracy on a test set. The CNN model achieved an impressive overall accuracy of 99.1% in identifying Early Blight, Late Blight, and Healthy potato leaf types. This research showcases the potential of deep learning methods in accurately categorising potato diseases, offering a reliable solution for disease identification crucial for agricultural sustainability and financial viability.

Shukla et al. introduced a Convolutional Neural Network (CNN)--based model for the early detection of potato leaf diseases [15]. Their study included image segmentation for feature extraction and a novel CNN technique for disease identification. With a dataset of 2250 images, their model achieved an impressive accuracy of 99.76%. They also developed a user-friendly web application for easy disease identification by young farmers, facilitating accurate pesticide dispensation. This work signifies a valuable advancement in leveraging deep learning for agricultural disease management.

Chapter 2

Problem Statement

The research focuses on a critical gap in existing literature regarding plant leaf disease analysis using computer vision techniques. While previous works have primarily relied on datasets such as Plant Village or images captured under controlled conditions with optimal lighting, proper orientation, and minimal background distractions, my research addresses the challenges of uncontrolled environments. These challenges include poor lighting conditions, varying angles of leaf images, background disturbances, and other environmental factors that can significantly impact the accuracy and reliability of disease detection algorithms.

The significance of this research problem lies in the need to develop robust and adaptive models capable of accurately identifying plant diseases even in real-world settings where conditions may not be ideal for image capture. Agricultural fields often present unpredictable situations, and relying solely on datasets captured under controlled environments may limit the applicability and effectiveness of disease detection systems in practical farm scenarios.

One significant observation from previous studies is the limited success achieved using Convolutional Neural Networks (CNNs) in such uncontrolled environments. While CNNs have shown remarkable performance in image recognition tasks under controlled conditions, their performance tends to degrade when applied to datasets with variations in lighting, angles, and background disturbances. This presents a significant challenge that needs to be addressed to advance the plant leaf disease analysis field.

2.1 RESEARCH GAPS

Previous research conducted in this field has utilised a Plant Village dataset. This dataset comprises high-quality images of potato leaves captured on a uniform, greyish-black background, offering a consistent and uniform appearance of the leaves. Additionally, the dataset has images taken under identical lighting conditions, viewing angles, and leaf orientations, thus providing the machine learning and deep learning models with the necessary inputs to comprehend the leaf patterns, edges, and diseases. This allows the models to detect and classify diseases more accurately and efficiently.

In December 2023, Shabrina et al. [2] presented an innovative dataset of potato leaf disease images captured in an uncontrolled environment. The dataset comprises diverse images of potato leaf disease captured under various uncontrolled environmental conditions. The environmental conditions make it challenging to identify patterns amidst the background noise, such as blurry photos and poor leaf orientation. Due to these factors, the original paper reported a maximum test accuracy of 73% using EfficientNet. This dataset contains valuable information that can assist researchers in developing more accurate and reliable models for identifying and diagnosing potato leaf diseases.

2.2 RESEARCH MOTIVATION

This research is motivated by recognising the limitations in current plant leaf disease analysis methodologies, particularly in the context of potato leaf diseases. While previous studies have predominantly relied on datasets like Plant Village, which offer controlled and standardised images conducive to accurate disease detection, there is a pressing need to address the challenges posed by uncontrolled environmental conditions in real-world agricultural settings.

Shabrina et al. introduced a novel dataset in December 2023 featuring potato leaf disease images captured in uncontrolled environments, which catalyses this research. The dataset's diversity and complexity, including variations in lighting, background noise, leaf orientation, and image quality, present a realistic representation of the challenges farmers and agronomists face in disease identification and diagnosis. The reported maximum test accuracy of 73% using EfficientNet indicates the difficulty of the task and underscores the opportunity for improvement in disease detection methodologies.

This research addresses critical gaps in the existing literature by focusing on this newly introduced dataset and its specific challenges. The motivation lies in enhancing machine learning and deep learning models' accuracy, robustness, and reliability for potato leaf disease analysis in uncontrolled environments. By developing innovative algorithms, preprocessing techniques, and model architectures tailored to handle the complexities of uncontrolled environmental conditions, this research seeks to advance the field of precision agriculture and contribute towards more effective disease management strategies for potato crops.

The ultimate goal is to bridge the gap between laboratory-based studies and real-world applications, empowering farmers with advanced tools and technologies to assist them in timely disease detection, proactive intervention, and sustainable agricultural practices. This research motivation underscores the importance of leveraging cutting-edge datasets and methodologies to address the evolving challenges in plant disease analysis and support global food security efforts.

2.3 RESEARCH CHALLENGES

The research challenges in addressing the uncontrolled environment potato leaf dataset are multifaceted and require innovative solutions. The dataset's poor quality is a significant obstacle in identifying and classifying disease symptoms in potato leaves. Images captured in uncontrolled environments may suffer from blur, low resolution, and inconsistent lighting. These factors can make it difficult to accurately identify and classify disease symptoms, which can negatively impact the effectiveness of the dataset.

The next significant challenge is the imbalance in the dataset. Imbalance occurs when particular sets or disease categories are overrepresented or underrepresented in the dataset, leading to biased model performance. To train robust and unbiased machine learning models, it is essential to balance the dataset by collecting diverse images that represent different disease severities, leaf orientations, and environmental conditions.

Another major obstacle in potato leaf disease analysis is adapting the YOLOv8 algorithm, currently the best object detection algorithm, to cater to the unique demands of analysing potato leaf disease in unpredictable settings. This necessitates adjusting the model's structure, optimising the hyperparameters, and applying data augmentation techniques to enhance the model's generalisation and performance on real-world datasets. Also, the lack of available literature and research studies that focus on the uncontrolled environment of the potato leaf dataset introduces complexity to the research landscape.

2.4 RESEARCH CONTRIBUTIONS

The research focuses on improving potato leaf disease detection using a Customized YOLOv8 Classifier and its different variants. We specifically worked with a dataset that mimics real-world conditions, which can be challenging due to varying environments and image quality issues.

1. A significant part of our contribution lies in preparing the dataset for analysis. We utilised different techniques to enhance the images before training the models. These included histogram equalisation, colour space conversion, and image denoising techniques.

2. We experimented with five different versions of the YOLOv8 Classifier on our enhanced images. By comparing these models, we gained insights into their strengths and weaknesses for this specific task. This information is essential for selecting the most suitable model for similar projects in the future.

3. Additionally, we developed a customised version of the YOLOv8 Classifier. Our customised YOLOv8 Classifier incorporates a Spatial Transformer Network (STN) for adaptive feature map transformation, enhancing disease spot detection accuracy. Additional convolutional and linear layers were added to capture complex patterns, culminating in improved classification performance.

Chapter 3

Proposed System and Methodology

The research work has four main components, which are as follows:

a) Data preprocessing: The images were divided into 70:15:15 ratios for training, validation, and testing. The image preprocessing methods used include denoising (Median and Gaussian Blur), Histogram Equalizations (HE, CLAHE), and colour space conversions (HSV).

b) Model training: Transfer learning was performed on non-pre-processed and pre-processed datasets on the nano, minor, medium, large, and extra-large variants of the YOLOv8 classifier.

c) Customized YOLOv8 (YOLOv8_STN): A customised YOLOv8 classifier with a Spatial Transformer Network and additional convolution layers was developed. Both the non-pre-processed and pre-processed images were trained, validated, and tested, and a performance comparison was performed.

d) Evaluation: The dataset's overall classification ability was evaluated using metrics like accuracy, precision, recall, and F1 score, with a preference for the F1 score due to class imbalance.

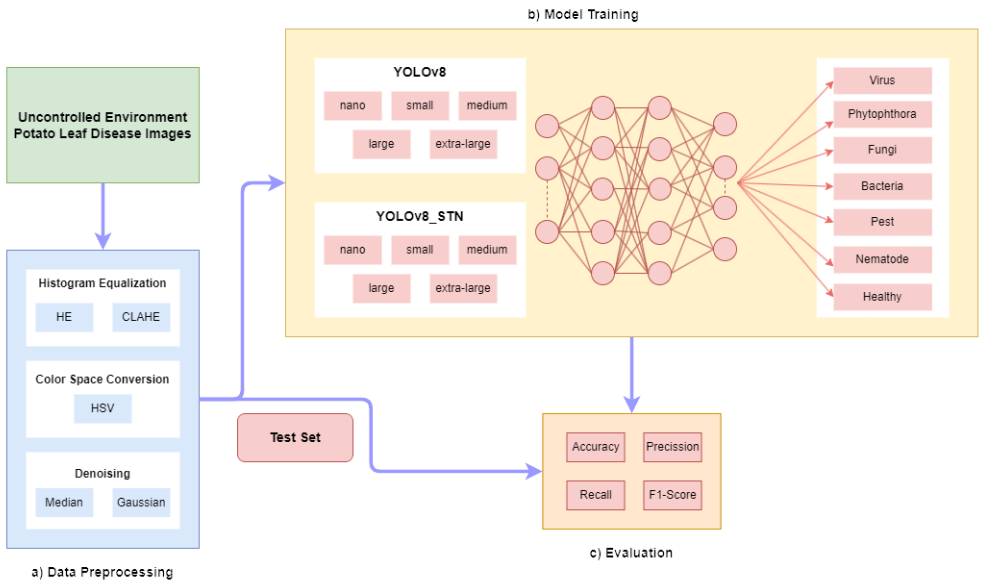


Figure 1: Proposed System

3.1 DATA PREPROCESSING

The dataset has been preprocessed using different kinds of Denoising, Color Space Conversions, and Hue Value Saturations. The following section explains each of them.

3.1.1 CLAHE

Contrast Limited Adaptive Histogram Equalization (CLAHE) is a sophisticated image preprocessing technique widely employed in plant leaf disease analysis to mitigate the challenges associated with varying illumination conditions, contrast disparities, and texture complexities in leaf images. The CLAHE algorithm partitions the input image into smaller, non-overlapping tiles. For each tile, a histogram of pixel intensities is computed, capturing the local contrast characteristics within that specific region. Unlike conventional histogram equalisation methods that apply a global transformation to the entire image, CLAHE applies a contrast enhancement process within each tile, ensuring the enhancement is adaptive and limited to the local context.

One of the critical aspects of CLAHE is its incorporation of a contrast limitation mechanism, which prevents excessive amplification of noise and artefacts that may arise during the histogram equalisation process. This is achieved by constraining each tile's cumulative distribution function (CDF), thus controlling the extent of contrast enhancement while maintaining the overall image quality. Furthermore, CLAHE incorporates an interpolation scheme to ensure smooth transitions between neighbouring tiles, which is essential for preserving the natural appearance of the image and avoiding discontinuities that could distort the analysis results.

For the plant leaf disease analysis, CLAHE plays a pivotal role in preprocessing leaf images before they undergo further study, such as disease detection, classification, and segmentation. By enhancing the local contrast and mitigating illumination variations, CLAHE significantly improves the visibility of subtle features, such as disease symptoms, textures, and patterns, which are crucial for accurate disease diagnosis. Also, CLAHE helps standardise the image quality across different samples, making the subsequent analysis more robust and reliable.

CLAHE's adaptive nature allows it to adapt to different types of plant leaves, varying lighting conditions, and image acquisition settings, making it a versatile and widely applicable preprocessing technique. Its effectiveness has been demonstrated in various studies and applications within plant pathology, where precise and reliable image analysis is essential for effective disease management and crop protection strategies. CLAHE stands as a cornerstone in plant leaf disease analysis, offering advanced capabilities in image enhancement and facilitating more accurate and insightful disease assessment and diagnosis.

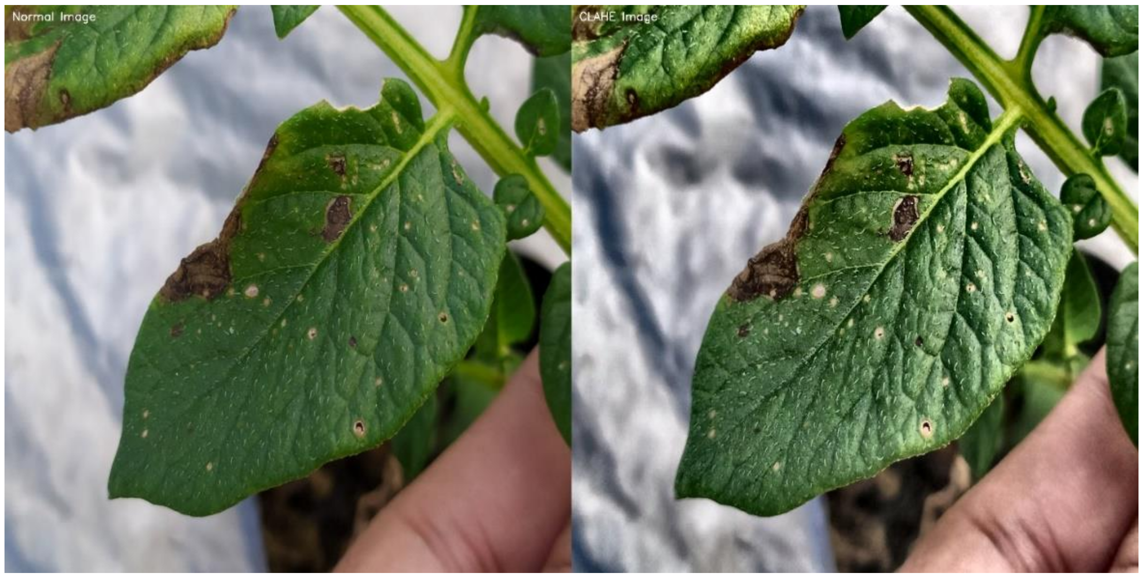


Figure 2: Non-preprocessed image and CLAHE image

3.1.2 HSV

Hue, Saturation, and Value (HSV) image preprocessing is a sophisticated technique used extensively in plant leaf disease analysis to enhance image quality and extract meaningful features for disease diagnosis. The HSV colour space represents colours in terms of hue (the colour itself), saturation (the intensity of the colour), and value (the brightness or lightness of the colour). This representation allows for more effective manipulation and analysis of colour information, making it particularly suitable for detecting subtle changes and variations in plant leaves indicative of disease symptoms.

In HSV image preprocessing for plant leaf disease analysis, the first step often involves converting the RGB (Red, Green, Blue) image into the HSV colour space. This conversion separates the colour and intensity components, providing a more intuitive representation of the image's colour information. Once in the HSV colour space, various preprocessing techniques can be applied to improve image quality and enhance features relevant to disease analysis.

One standard preprocessing operation is a contrasting enhancement, which can be achieved by adjusting the value component of the HSV image. By applying techniques such as histogram equalisation or contrast stretching to the value channel, the overall contrast of the image can be improved, making it easier to distinguish between healthy and diseased areas on the leaf.

Another critical aspect of HSV preprocessing is colour normalisation or standardisation. Since lighting conditions and camera settings can vary significantly between image acquisitions, normalising the colour information helps to reduce the effects of these variations and ensures consistency across different images. Techniques such as white balancing or colour calibration can achieve colour normalisation in HSV space, enhancing the reliability and accuracy of subsequent disease analysis algorithms.

Moreover, HSV preprocessing can involve segmentation techniques to isolate regions of interest, such as diseased areas or specific leaf structures. By thresholding, the hue, saturation, or value components based on predefined criteria or using machine learning-based segmentation algorithms, meaningful features related to disease symptoms, such as discolouration, lesions, or texture irregularities, can be extracted.

HSV image preprocessing is crucial in plant leaf disease analysis. It improves image quality, standardises colour information, enhances contrast, and extracts relevant disease diagnosis and classification features. Its versatility and effectiveness make it a valuable tool for improving plant pathology studies' accuracy, reliability, and disease management strategies.

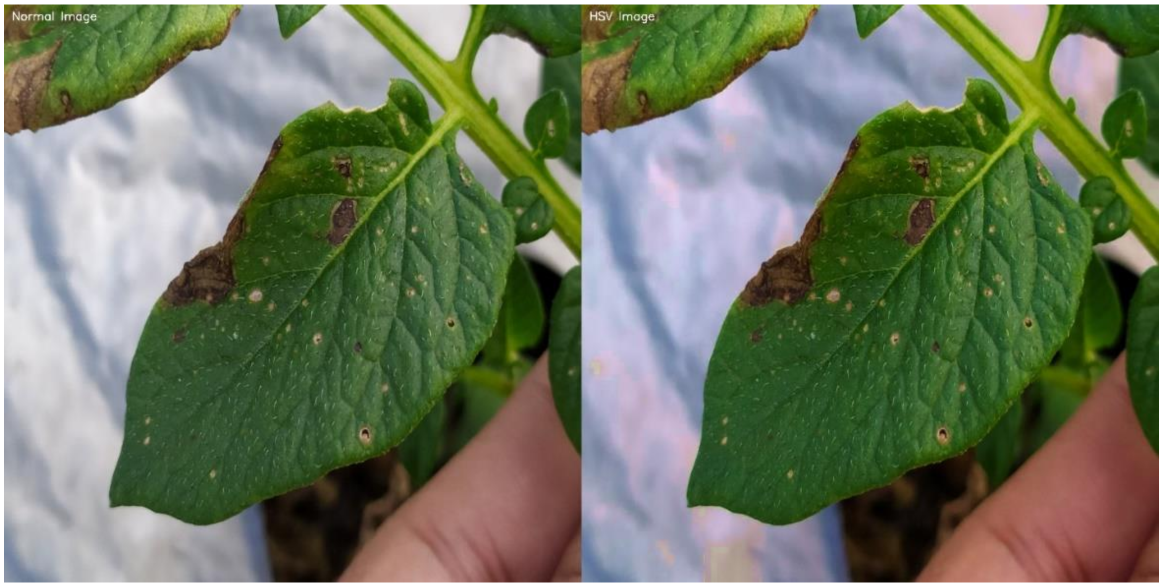


Figure 3: Non-preprocessed image and HSV image

3.1.3 HISTOGRAM EQUALIZATION

Histogram Equalization is a fundamental image preprocessing technique utilised extensively in plant leaf disease analysis to enhance image quality and improve the effectiveness of subsequent analysis tasks. This method operates by redistributing the pixel intensities of an image to achieve a more uniform histogram distribution, thereby enhancing the contrast and visibility of features within the image. In plant pathology, where subtle variations in leaf textures, colours, and disease symptoms are crucial for accurate diagnosis, histogram equalisation is vital in improving the distinguishability of such features.

The histogram equalisation process involves computing the input image's histogram, representing the frequency of occurrence of each pixel intensity level. Next, a cumulative distribution function (CDF) is calculated based on the histogram, mapping each pixel intensity to a new value derived from the CDF. This mapping redistributes the pixel values so regions with lower contrast or intensity variations are stretched. In comparison, areas with higher contrast are compressed, leading to an overall improvement in image contrast and visibility of details.

For the plant leaf disease analysis, histogram equalisation can effectively enhance the visibility of disease symptoms such as discolourations, lesions, or texture irregularities that may be challenging to detect in the original image due to varying lighting conditions, camera settings, or leaf characteristics. By equalising the histogram, subtle features associated with different disease stages or severity levels become more pronounced, aiding in accurately identifying and classifying plant diseases.

However, it's essential to note that histogram equalisation may also amplify noise and artefacts present in the image, especially in regions with low contrast or uniform textures. Variants of histogram equalisation techniques, such as Contrast Limited Adaptive Histogram Equalization (CLAHE), are often employed to address this issue. CLAHE restricts the contrast enhancement within localised regions or tiles, preventing excessive noise amplification and maintaining a balanced enhancement across the image.

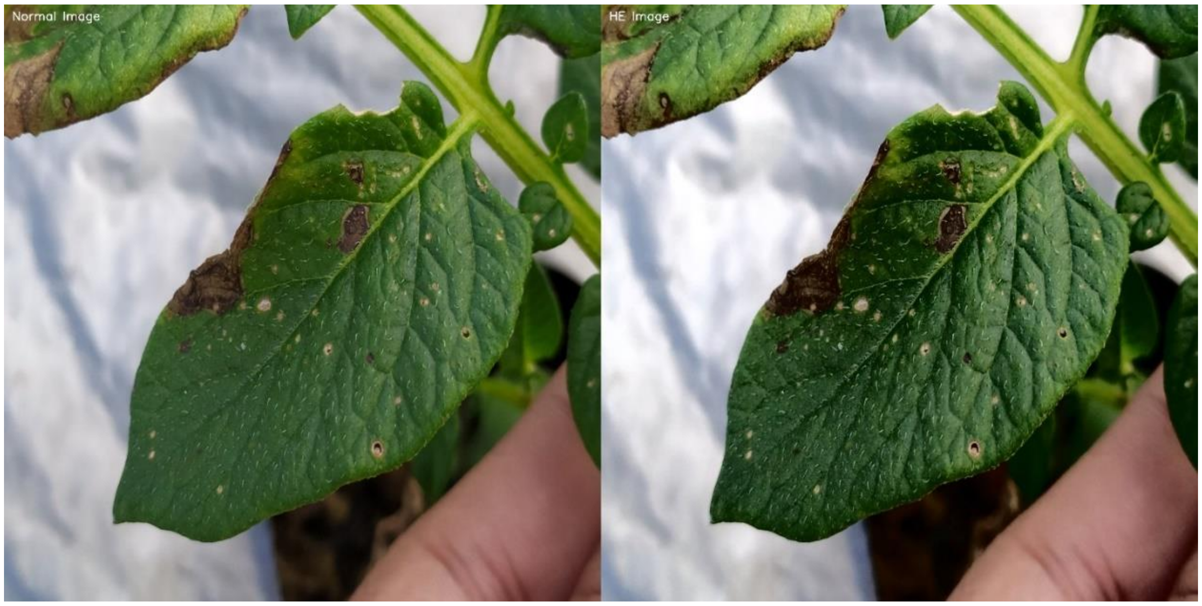


Figure 4: Non-preprocessed image and Histogram Equalized image

3.1.4 MEDIAN FILTERING

Median filtering is a popular image processing technique used in various applications, including plant leaf disease analysis, to reduce noise and enhance image quality. In plant pathology, where precise identification and analysis of disease symptoms on leaf images are crucial, median filtering significantly improves the accuracy of subsequent analysis algorithms.

The median filter replaces each pixel in the image with the median value of its neighbouring pixels within a defined window or kernel size. Unlike mean filtering, which computes the average of pixel values, median filtering uses the median value, making it robust against outliers and preserving edges and fine details in the image. This property is particularly beneficial in plant leaf images, where noise can arise from various sources such as camera sensors, environmental conditions, or image acquisition processes.

One of the primary advantages of median filtering in plant leaf disease analysis is its ability to effectively remove different types of noise, including salt-and-pepper and Gaussian noise, without significantly blurring the image or compromising essential details. Salt-and-pepper noise, characterised by randomly occurring bright and dark pixels, can obscure disease symptoms and interfere with automated analysis algorithms. Median filtering effectively eliminates noise while preserving the integrity of the underlying image structure.

Furthermore, median filtering is computationally efficient and straightforward, making it suitable for daily real-time applications and large-scale image processing tasks in plant pathology research. By reducing noise and enhancing image clarity, median filtering prepares leaf images for more accurate feature extraction, segmentation, and disease classification processes, ultimately contributing to improved disease diagnosis and management strategies in agriculture.

However, it's essential to consider the filter window or kernel size choice when applying median filtering, as more oversized windows may lead to increased smoothing and potential loss of fine details. In comparison, smaller windows may not effectively remove all noise components. Therefore, parameter tuning and experimentation are often necessary to determine the optimal filter settings based on the leaf images' specific characteristics and the desired noise reduction level. Overall, median filtering is a valuable preprocessing step in plant leaf disease analysis, enhancing image quality and facilitating accurate disease detection and characterisation.

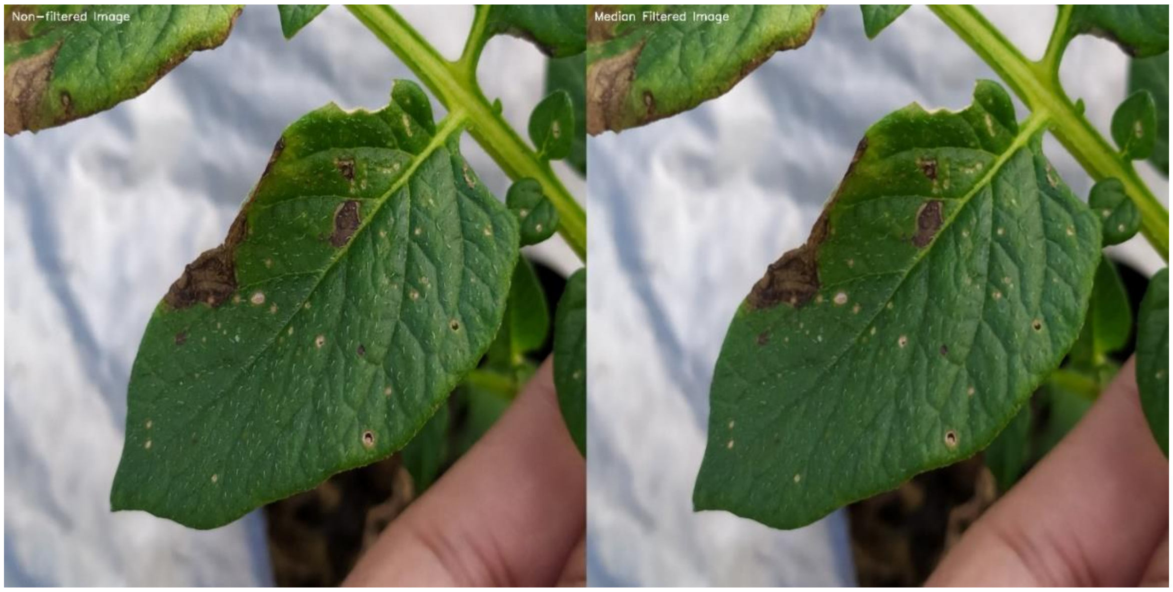


Figure 5: Non-preprocessed image and Median Filtered image

3.1.5 GAUSSIAN FILTERING

Gaussian filtering is a widely used image processing technique that plays a crucial role in plant leaf disease analysis by improving image quality, reducing noise, and enhancing feature extraction for accurate disease diagnosis. The method derives its name from the Gaussian distribution, a mathematical function that defines the kernel or filter applied to the image.

A convolution operation is performed between the input image and a Gaussian kernel in Gaussian filtering. The Gaussian kernel is characterised by a bell-shaped curve with higher weights at the centre and gradually decreasing weights towards the edges. This weighting scheme ensures that pixels closer to the centre of the kernel have a more substantial influence on the resulting filtered pixel value, while pixels farther away contribute less. This property allows Gaussian filtering to effectively smooth the image while preserving essential details and edges, making it particularly suitable for noise reduction in plant leaf images.

One key advantage of Gaussian filtering in plant leaf disease analysis is its ability to mitigate various types of noise, including Gaussian noise, which is common in images due to sensor limitations, environmental factors, or image acquisition processes. Gaussian noise appears as random variations in pixel intensities and can obscure disease symptoms and affect the accuracy of automated analysis algorithms. By applying Gaussian filtering, the noise is smoothed out, resulting in a cleaner and more visually appealing image.

The Gaussian-filtered images provide more suitable input for these algorithms by reducing irrelevant noise artefacts and enhancing the visibility of disease-related features such as lesions, discolourations, or texture irregularities.

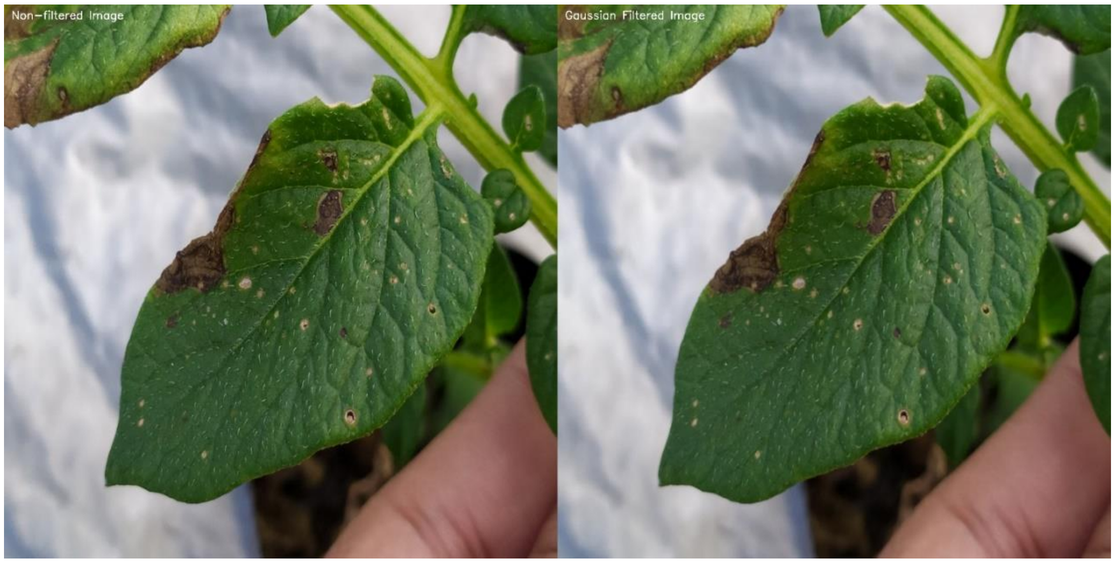


Figure 6: Non-preprocessed image and Gaussian Filtered image

3.2 YOLO

You Only Look Once (YOLO) is an advanced object detection algorithm that has set new benchmarks for computer vision speed, accuracy, and efficiency. Initially introduced by Joseph Redmon et al., YOLO represents a paradigm shift in object detection methodologies, offering a real-time, end-to-end approach that eliminates the need for complex pipelines with multiple stages.

The fundamental principle of YOLO revolves around dividing the input image into a grid and performing object detection directly at the grid cell level. Unlike traditional algorithms that propose regions of interest and classify objects within those regions, YOLO simultaneously predicts bounding boxes and class probabilities in a single pass through a convolutional neural network (CNN). This unique approach significantly accelerates the detection process and improves the accuracy of object localisation and classification.
The architecture of YOLO typically consists of a deep CNN backbone, such as Darknet or ResNet, followed by detection layers responsible for generating predictions. These detection layers predict bounding box coordinates (centre coordinates, width, height), confidence scores indicating the accuracy of the bounding box, and class probabilities for different object categories. YOLO employs anchor boxes to handle objects of various sizes and aspect ratios, enabling it to effectively detect small objects and objects at different scales.

One of YOLO's key strengths is its speed, which allows it to achieve real-time performance on standard hardware. This makes YOLO ideal for applications requiring rapid object detection, such as video analysis, autonomous vehicles, robotics, and surveillance systems. Despite its speed, YOLO maintains high accuracy, outperforming many traditional algorithms and even some slower state-of-the-art detectors.

Over time, YOLO has undergone several iterations and improvements, leading to versions like YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, and YOLOv8. These iterations introduce enhancements such as feature pyramid networks (FPN) to capture multi-scale features, improved training strategies, and architectural optimisations to boost performance further and address limitations like handling small objects and achieving better generalisation.

YOLO, which stands for You Only Look Once, is an innovative object detection algorithm known for its real-time performance and high accuracy. The architecture of YOLO consists of several vital components that work together to detect objects in images efficiently.

1. Input Image Division: The input image is divided into a grid of cells. Each cell predicts bounding boxes and object classes for objects within its boundaries.

2. Convolutional Neural Network (CNN) Backbone: YOLO typically uses a deep CNN backbone, such as Darknet or ResNet, to extract features from the input image. These features are then used for object detection.

3. Anchor Boxes: YOLO utilises anchor boxes to handle objects of different sizes and aspect ratios within each grid cell. Anchor boxes are predefined bounding boxes with specific dimensions and shapes that the model uses to predict accurate bounding box coordinates.

4. Detection Layer: YOLO's architecture includes detection layers that predict bounding box coordinates (centre coordinates, width, height), confidence scores indicating the accuracy of the bounding boxes, and class probabilities for different object categories. Each grid cell predicts multiple bounding boxes and their associated probabilities.

5. Non-Maximum Suppression (NMS): After predicting multiple bounding boxes for each object, YOLO applies Non-Maximum Suppression to filter out redundant or overlapping detections. This step ensures that each object is detected only once with the most accurate bounding box.

6. Output: YOLO's final output is a set of bounding boxes, confidence scores, and class probabilities for all detected objects in the input image.

YOLO's architecture enables object detection in a single pass through the network, making it extremely fast compared to traditional methods involving multiple processing stages. This real-time capability makes YOLO well-suited for applications where speed is critical, such as video analysis, autonomous vehicles, and surveillance systems. Additionally, YOLO's ability to handle objects of different sizes and its high accuracy have made it a popular choice among researchers and practitioners in the computer vision community.

3.3 YOLOv8 CLASSIFIERS

YOLOv8 is an advanced AI framework that supports multiple computer vision tasks, including detection, segmentation, OBB (oriented bounding boxes), classification, and pose estimation. Each of these tasks serves a different purpose and caters to various use cases within the field of computer vision. One of the critical functionalities of YOLOv8 is its ability to perform image classification, which involves categorising an entire image into predefined classes based on its content. To achieve this, YOLOv8 leverages a variant of the EfficientNet architecture, known for its efficiency and effectiveness in handling classification tasks.

In image classification, the model's output consists of a single class label and a confidence score indicating the model's certainty in its prediction. This task is beneficial when the objective is to determine an image's class without needing precise object localisation or shape information. YOLOv8 provides pre-trained models for image classification with different sizes and complexities to suit various computational requirements and performance needs.

The pre-trained Classify models in YOLOv8 are categorised based on their size, including YOLOv8n-cls, YOLOv8s-cls, YOLOv8m-cls, YOLOv8l-cls, and YOLOv8x-cls. These models vary in terms of their accuracy (top1 and top5), computational speed (measured in milliseconds for CPU ONNX and A100 TensorRT), model parameters (M), and computational complexity (FLOPs at 640 pixels).

These pre-trained Classify models in YOLOv8 benefit various applications, including image recognition, content-based filtering, and automated classification tasks. They balance accuracy, computational efficiency, and model complexity, making them suitable for deployment in real-world scenarios where fast and accurate image classification is required. YOLOv8's integration with the latest Ultralytics release also ensures seamless model downloading and user accessibility, enhancing its usability and convenience in computer vision applications.

3.4 SPATIAL TRANSFORMER NETWORK

Spatial Transformer Networks (STNs) are sophisticated neural network modules that enable deep learning models to learn and apply spatial transformations autonomously. They have three main components: the localisation network, the grid generator, and the sampler. The localisation network serves as the brain of the STN, taking an input image or feature map and predicting transformation parameters such as translation (tx, ty), rotation (θ), scaling (sx, sy), and shearing (ϕx, ϕy). These parameters define how the input data should be geometrically transformed to achieve the desired output. The grid generator then uses these parameters to generate a sampling grid, specifying where to sample values from the input data to create the transformed output. Finally, the sampler uses this grid to transform the spatial, sampling the input data according to the specified parameters.

STNs offer several advantages in the realm of computer vision and deep learning. Firstly, they enable models to learn spatial transformations directly from the data, eliminating the need for manual feature engineering or explicit supervision for these transformations. This adaptability makes STNs highly versatile and applicable to various tasks, including image classification, object detection, image registration, and image generation. Secondly, STNs enhance model robustness by allowing them to handle variations in input data due to different orientations, scales, or positions. This robustness is crucial for real-world applications where input data may exhibit diverse spatial characteristics. Additionally, STNs can improve the efficiency and accuracy of deep learning models by focusing on relevant spatial regions and reducing computational overhead.

In practical applications, STNs have been successfully integrated into various neural network architectures, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), enhancing their spatial transformation capabilities. By enabling models to learn and apply spatial transformations end-to-end, STNs contribute significantly to advancing computer vision systems, allowing them to handle complex spatial tasks more effectively and achieve superior performance in diverse real-world scenarios.

3.5 YOLOv8_STN – OUR PROPOSED ARCHITECTURE

Our research paper introduces a novel architecture that extends the YOLOv8. Our model, YOLOv8_STN, incorporates a Spatial Transformer Network (STN) module to transform feature maps, enhancing spatial information processing dynamically. The input image is first processed by the YOLOv8 backbone, which extracts high-level features and performs object detection tasks. The output feature maps from the YOLOv8 backbone are fed into the STN module, consisting of a localisation network followed by fully connected layers (FC_loc) to learn spatial transformations. The learned transformation matrix (θ) generates an affine grid, which is then applied to the input feature maps through grid sampling, resulting in spatially transformed feature maps. These transformed feature maps undergo further processing through additional convolutional layers and custom blocks, enhancing their discriminative power and preparing them for final feature aggregation and predictions. The integration of STN into the YOLOv8 architecture enables the network to adjust and focus on relevant regions of the input dynamically, leading to performance improvements.

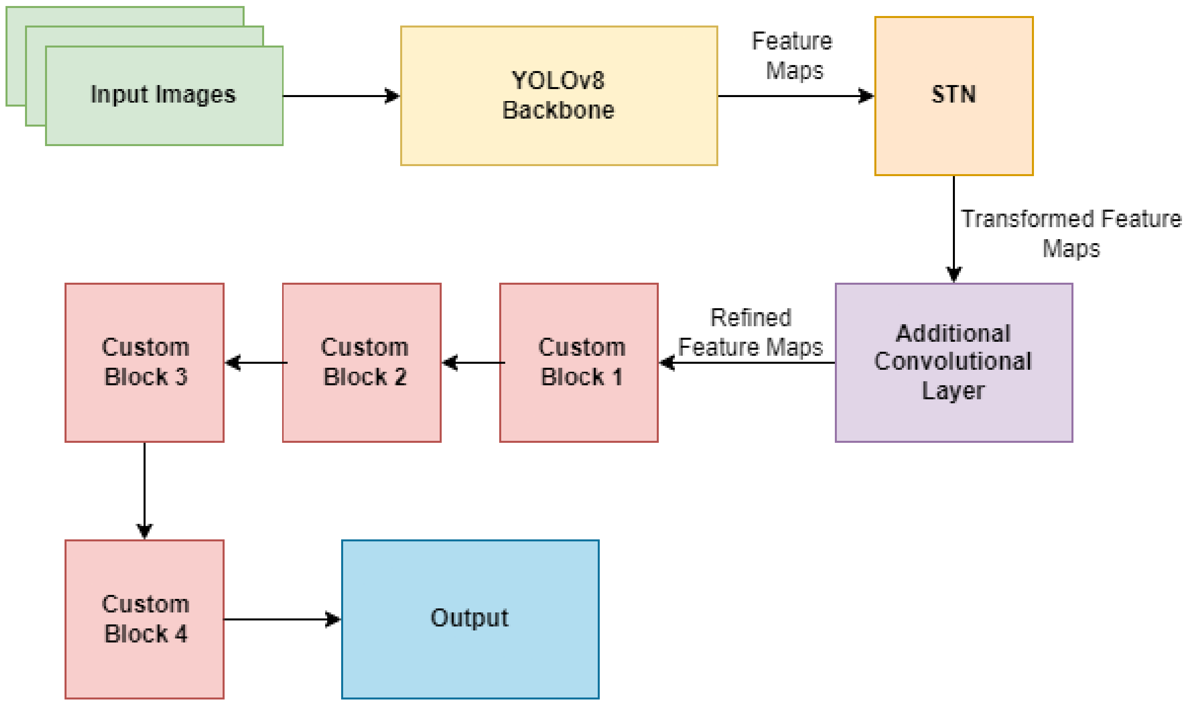


Figure 7: Overview of the proposed YOLOv8_STN Architecture

Let's discuss each block and its functions in detail.

3.5.1 SPATIAL TRANSFORMER BLOCK (STN)

The STN block within the YOLOv8_STN architecture is designed to dynamically adjust and focus on relevant regions within the input feature maps. It begins with a localisation network comprising two convolutional layers. The first convolutional layer has a kernel size of 7x7, operates on input channels specific to the data, and is followed by a 2x2 max-pooling operation with a stride of 2, effectively reducing the spatial dimensions of the feature maps. The subsequent convolutional layer in the localisation network has a kernel size 5x5, continuing to extract spatial features and refine the representation. The feature maps are then reshaped and processed through fully connected layers, including a linear transformation with specific input and output sizes, followed by a ReLU activation function. This STN block utilises learned parameters to generate affine transformations, facilitating spatial warping through affine grid generation and grid sampling operations.

3.5.2 ADDITIONAL CONVOLUTION BLOCK

The Additional Convolution (additional_conv) block follows the STN module and focuses on further feature extraction and refinement. It consists of multiple convolutional layers with specified kernel sizes, input channels, output channels, strides, and padding. For instance, the first convolutional layer within the additional_conv block has an input channel of 512, a kernel size of 3x3, a stride of 1, and padding of 1, followed by ReLU activation and batch normalisation with 1024 channels. Similarly, the subsequent convolutional layers maintain this structure to process and enhance the feature representations obtained from the STN module.

3.5.3 CUSTOM BLOCK 1

Custom Block 1 operates after the additional_conv block and aims to refine features further through dimensionality reduction and feature extraction. It begins with a convolutional layer having an input channel of 1024, a 1x1 kernel size for channel reduction to 512, and ReLU activation to introduce non-linearity. Batch normalisation follows to stabilise and normalise the feature activations, contributing to improved model convergence during training.

3.5.4 CUSTOM BLOCK 2

Custom Block 2 continues the feature refinement process with consecutive convolutional layers. The first convolutional layer within this block operates on 512 input channels, utilises a 3x3 kernel size, and preserves spatial dimensions with padding. ReLU activation and batch normalisation follow to enhance feature representations. Subsequently, another convolutional layer refines the features further, maintaining the structure of kernel size, activation, and normalisation to capture more complex and abstract features.

3.5.5 CUSTOM BLOCK 3

Custom Block 3 focuses on fine-grained feature extraction and noise reduction. It comprises a convolutional layer with parameters such as a 3x3 kernel size, 256 input channels, and ReLU activation for non-linearity. Batch normalisation stabilises the feature activations, ensuring robustness and efficiency in feature representation, which is particularly useful in object detection tasks.

3.5.6 CUSTOM BLOCK 4

Custom Block 4 involves a linear transformation layer followed by activation and normalisation. This block processes the features aggregated from previous layers, with the linear layer performing dimensionality reduction from 1000 to 512, introducing non-linearity with ReLU activation, and stabilising activations through batch normalisation. This process results in compact and informative feature representations conducive to accurate object detection predictions within the YOLOv8_STN architecture.

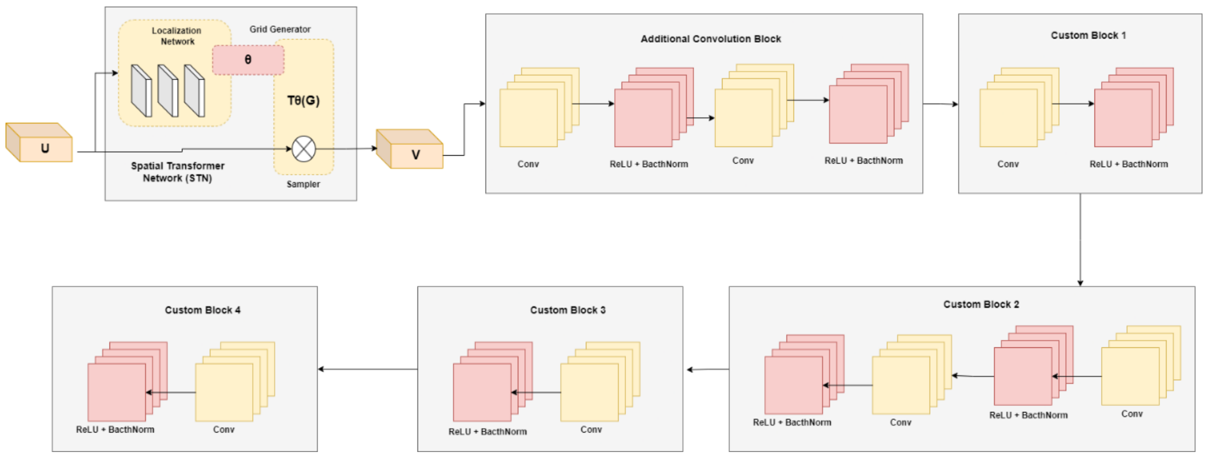


Figure 8: Detail view of the proposed YOLOv8_STN architecture

Chapter 4

Results and Discussions

This section presents the dataset, environmental setups, evaluation metrics, performance, and results of YOLOv8, YOLOv8_STN, and their variants, which were evaluated using various preprocessing techniques.

4.1 DATASET DESCRIPTION

This study employs the publicly accessible dataset "A Novel Dataset of Potato Leaf Disease in an Uncontrolled Environment" [2]. The dataset contains 3076 labelled images that belong to seven classes: virus, Phytophthora, Fungi, Bacteria, Pest, Nematode, and Healthy. Table 1 provides the number of samples available in each class.

Table 1: Distribution of instances for seven potato leaf categories

S. No.	Class	Number of Samples
1	Virus	532
2	Phytophthora	347
3	Fungi	748
4	Bacteria	569
5	Pest	611
6	Nematode	68
7	Healthy	201

4.2 ENVIRONMENTAL SETUP AND HYPERPARAMETERS

The NVIDIA Tesla T4 accelerator with 15 GB of RAM at Google Colab was utilised to perform this research work. The proposed experiments were implemented using Python and the PyTorch framework. To develop effective deep learning networks, it is crucial to optimise hyperparameters as they significantly impact the model's ability to learn and generalise from input data. Based on the study, it was found that the highest network convergence was achieved with a learning rate of 0.000714. Moreover, the study revealed that the AdamW optimiser yielded superior results. The batch size was set to 16. The hyperparameters are summarised in Table 2.

Table 2: Hyperparameters for models.

Parameter	Value
Learning Rate	0.000714
Optimizer	AdamW
Number of Epochs	10
Batch Size	16

4.3 EVALUATION METRICS

Assessing the performance and effectiveness of deep learning models is crucial, and evaluation metrics play a significant role in achieving this. These metrics enable the comparison and selection of the best model from various candidates. We evaluated the models on the Uncontrolled Environment Potato Leaf dataset using accuracy, precision, recall, and F1-score as demonstrated in equations (1) to (4). Although we mentioned all four evaluation metrics, the primary focus was on the F1-Score. This is because there was a class imbalance in the dataset, and accuracy can be misleading in such cases because it may favour the majority class and overlook the minority classes, leading to an inflated sense of performance. Instead, the F1 score is a more suitable metric for imbalanced datasets because it considers both precision and recall, giving equal importance to false positives (precision) and false negatives (recall). The F1 score is instrumental in balancing and minimising false positives and negatives. The performance of a classification model is thoroughly broken down in the confusion matrix; This is a tabular representation. It displays the total number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) for each class.

$$Accuracy = \frac{TP}{TP+TN+FP+FN} \tag{1}$$

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$F1 - score = \frac{2*(Precision*Recall)}{Precision+Recall} \tag{4}$$

4.4 PERFORMANCE ANALYSIS

This section analyses the performance of various YOLOv8 variants on the non-pre-processed and pre-processed Uncontrolled Environment Potato Leaf dataset. It also compares the performance of the YOLOv8 and the proposed YOLOv8_STN. A performance comparison between YOLOv8 and YOLOv8_STN has also been made on the pre-processed dataset to see if the pre-processing helps improve the performance of the proposed YOLOv8_STN. Finally, the section includes a performance comparison between the proposed YOLOv8_STN model and other CNN models on the Plant Village dataset, aiming to validate the robustness of the proposed model.

4.4.1 PERFORMANCE ANALYSIS OF YOLOV8 VARIANTS BETWEEN NON-PRE-PROCESSED DATASETS AND DIFFERENT PREPROCESSED DATASETS.

The five variants of YOLOv8, namely nano, small, medium, large, and extra-large, were implemented with different histogram equitisation (HE, CLAHE), colour space conversion technique (HSV), and denoising techniques (Median, Gaussian). Transfer Learning was performed, and an F1 score was utilised to check the performance of the models using the different pre-processing methods. Figure 9 shows the performance comparison using the F1-Score. It can be concluded from the bar chart that these preprocessing techniques helped in improving the performance of the YOLOv8 and its different variants. The models' accuracy, precision, recall, and F1 scores for all pre-processing techniques can be found in the tables provided in the Appendix.

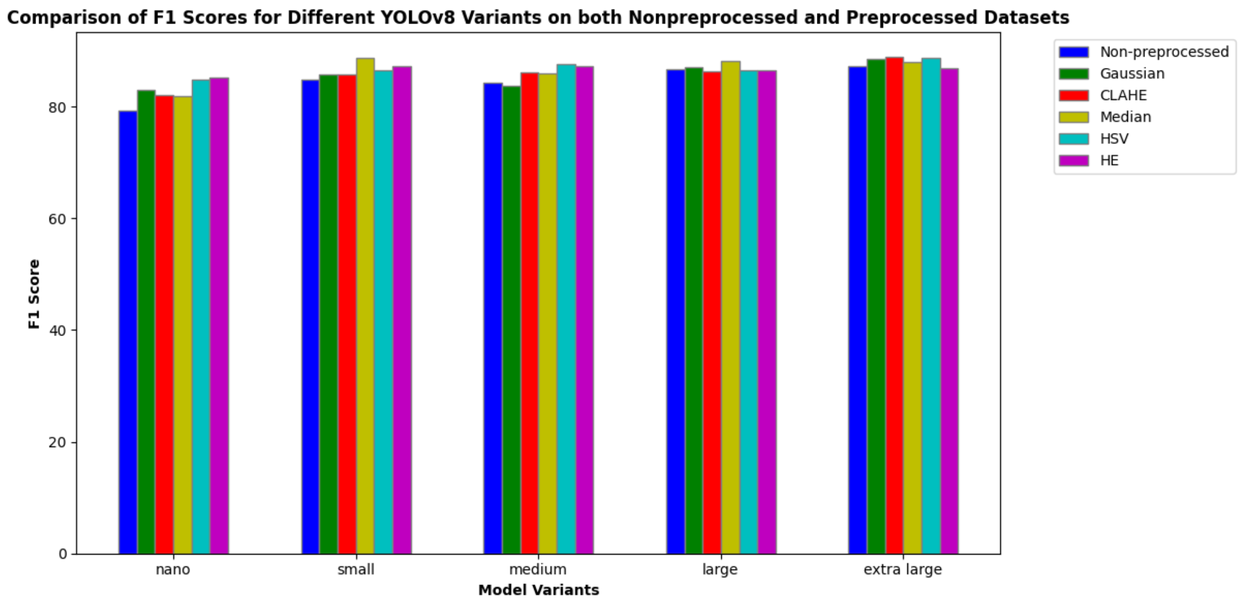


Figure 9: Comparison of F1 Scores for Different YOLOv8 Variants on both Non-pre-processed and Pre-processed Dataset

Table 3: Evaluation metrics for Different YOLOv8 Variants on both Non-pre-processed and Pre-processed Dataset

Variant	Non-pre-processed	Gaussian	CLAHE	Median	HSV	HE
nano	79.25	82.92	82.02	81.92	84.89	85.2
small	84.81	85.86	85.85	88.76	86.6	87.25
medium	84.27	83.74	86.06	86	87.71	87.33
large	86.75	87.11	86.41	88.13	86.47	86.53
extra large	87.26	88.63	88.85	88.03	88.77	86.8

4.4.2 PERFORMANCE ANALYSIS BETWEEN YOLOV8 AND THE PROPOSED YOLOV8_STN ON NON-PRE-PROCESSED DATASET

The evaluation clearly showed that YOLOv8_STN performed better than the YOLOv8 architecture it was based on. Integrating the Spatial Transformer Network (STN) enabled dynamic spatial transformations and region-based attention, improving object localisation—additional convolutional layers and custom blocks further refined feature representations, enhancing object model accuracy and robustness.

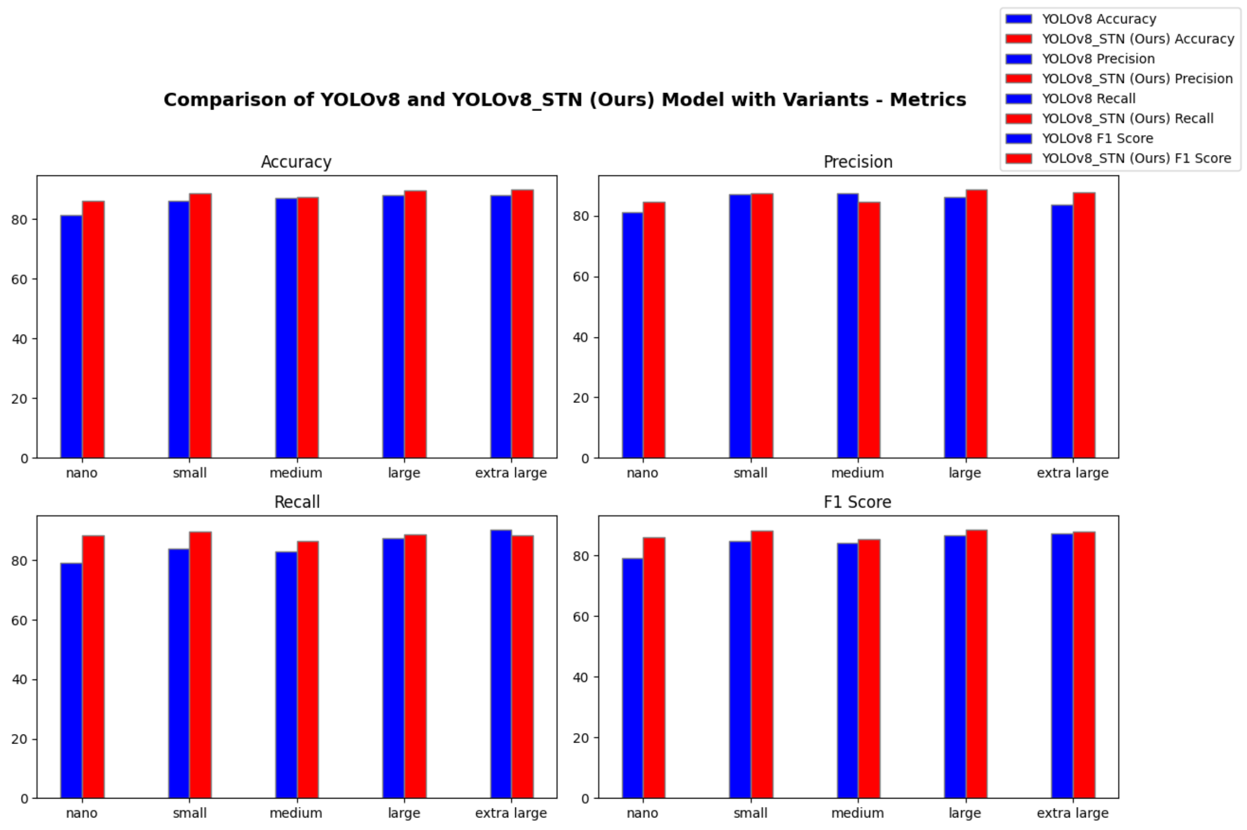


Figure 10: Comparison of YOLOv8 Variants with YOLOv8_STN variants

Table 4: Evaluation metrics of YOLOv8 Variants on non-pre-processed dataset

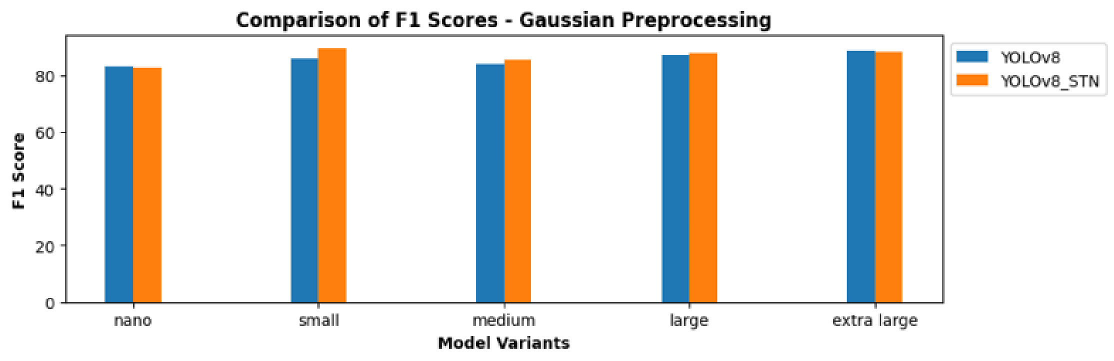
Variant	Accuracy	Precision	Recall	F1
nano	81.54	81.09	79.05	79.25
small	86.26	87.27	83.95	84.81
medium	87.12	87.38	83.08	84.27
large	88.19	86.34	87.53	86.75
extra large	88.19	83.62	90.49	87.26

Table 5: Evaluation metrics of YOLOv8_STN Variants on non-pre-processed dataset

Variant	Accuracy	Precision	Recall	F1
nano	86.26	84.56	88.27	86.01
small	88.84	87.39	89.62	88.31
med	87.55	84.7	86.57	85.31
large	89.69	88.85	88.77	88.71
extra large	90.12	87.88	88.56	87.82

4.4.3 PERFORMANCE ANALYSIS BETWEEN YOLOV8 AND THE PROPOSED YOLOV8_STN ON PREPROCESSED DATASETS

As the previous section shows that the YOLOv8_STN outperformed YOLO_v8, we will compare the performance of both models' variants on different preprocessed datasets.



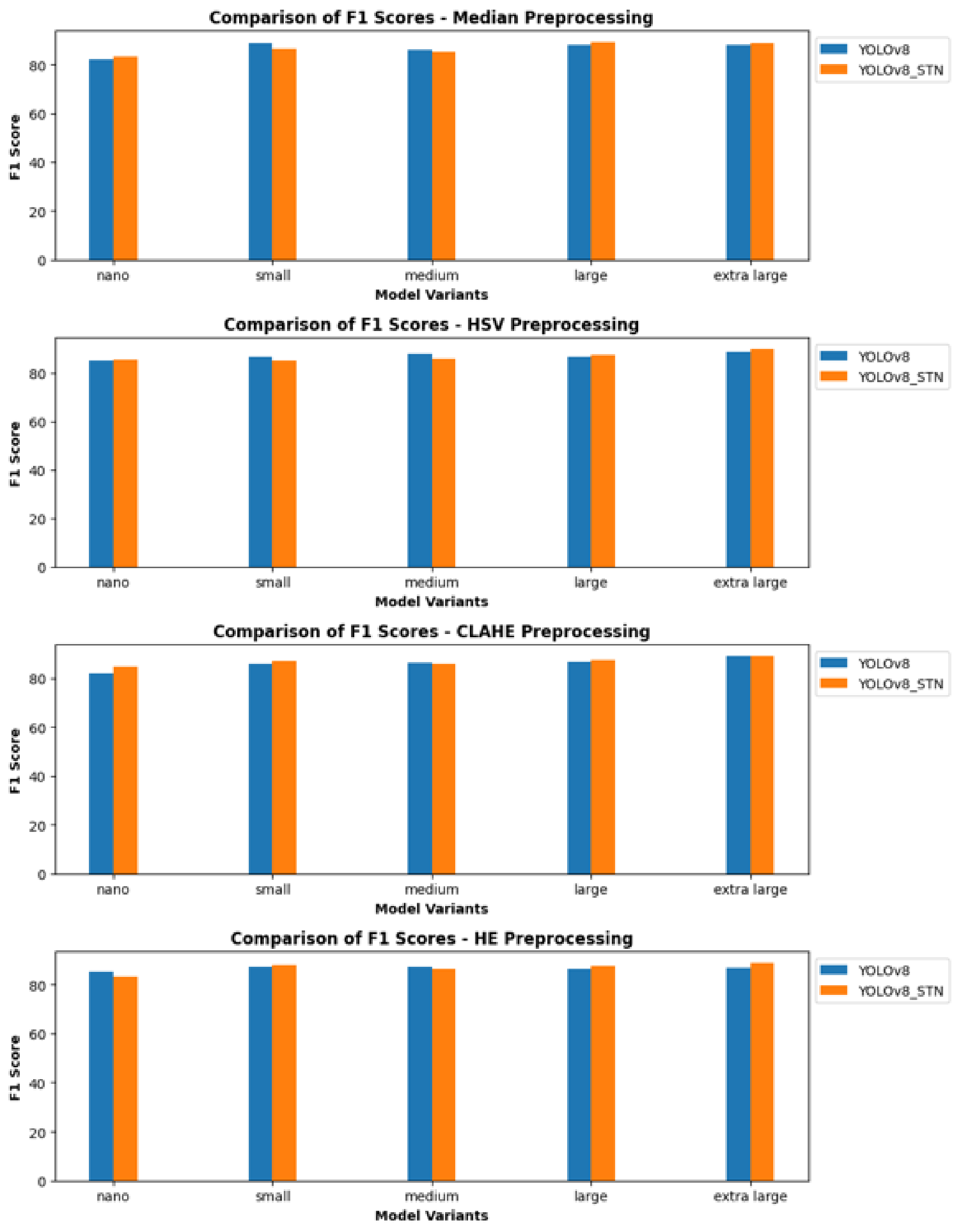


Figure 11: F1-Score comparison of YOLOv8 variants with YOLOv8_STN variants on pre-processed datasets

As evident in Figure 11, the preprocessed datasets performed better on the YOLOv8_STN than the YOLOv8. Out of all the preprocessing techniques, the Hue Saturation Value (HSV) was found to have the highest F1-Score. Coincidentally, it also has the highest accuracy, recall, and precision among all the preprocessed techniques and variants. The detailed values of all the preprocessed and non-preprocessed evaluation metrics can be seen in the appendix. Figure 12 shows the normalised confusion metrics of the HSV pre-processed extra-large YOLOv8_STN model. Figure 13 shows the model training curves for the best-performing HSV pre-processed extra-large YOLOv8_STN model. Table 6 shows the top-performing YOLOv8_STN variants for each pre-processing technique.

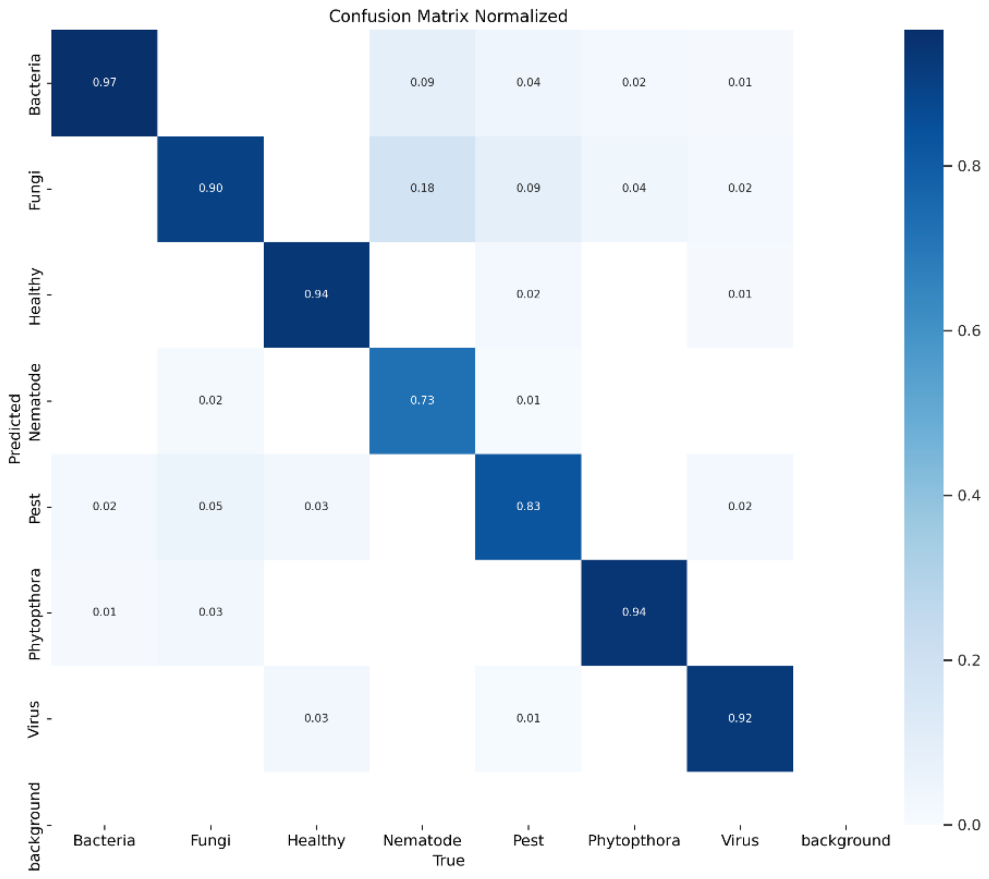


Figure 12: Normalized Confusion matrix of the best performing YOLOv8 Extra-large Variant on HSV preprocessing

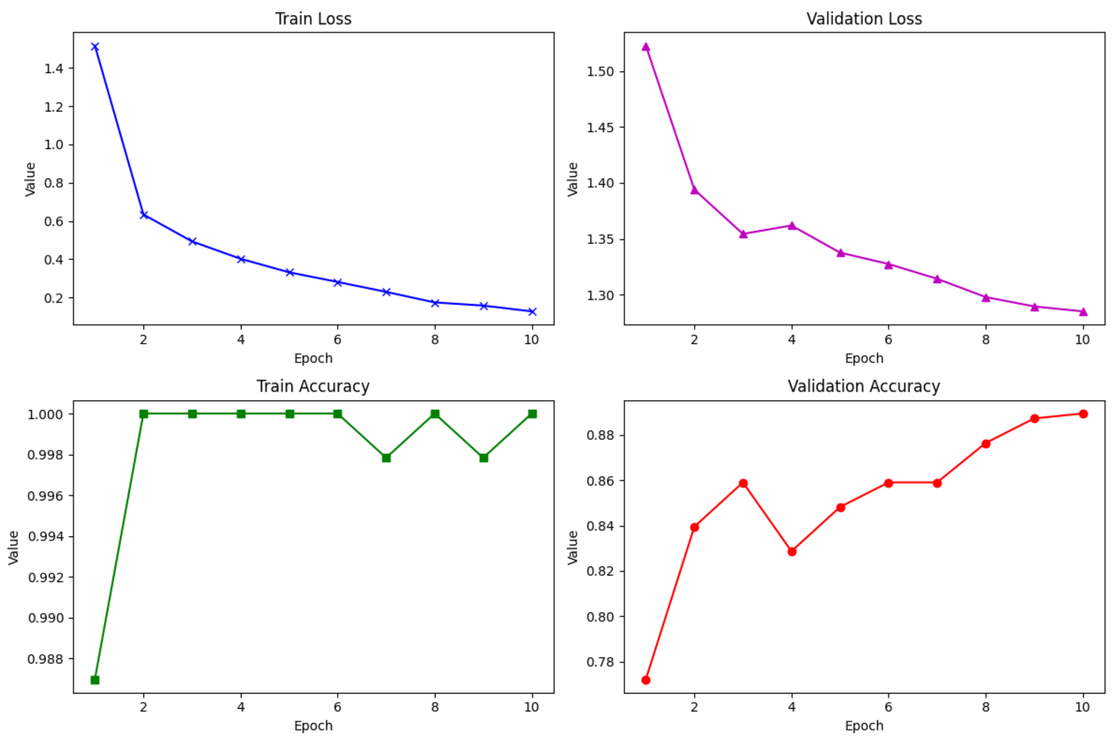


Figure 13: Model Training Curves of the best performing Yolov8 Extra-large Variant on HSV preprocessing

Table 6: Best performing YOLOv8_STN variants on each preprocessing technique.

Preprocessing-Variant	Accuracy	Precision	Recall	F1-Score
Gaussian- small	89.91	89.01	90.46	89.52
CLAHE- extra large	90.27	90.01	88.26	89.12
Median- large	89.27	88.56	90.56	89.34
HE – extra large	89.15	89.16	89.14	88.96
HSV – extra large	**90.81**	**92.97**	**87.93**	**90.02**

4.4.4 COMPARISON OF THE PROPOSED YOLOV8 WITH OTHER STAT-OF-THE-ART CNN MODELS ON PLANT VILLAGE DATASET

Our proposed model demonstrated promising performance on the Uncontrolled Environment Potato Leaf Disease dataset. To assess its robustness and generalisation capabilities, we are now evaluating it using the widely recognised Potato Leaf dataset – "The Plant Village Dataset." We will then conduct a comparative analysis with state-of-the-art CNN models to gauge the effectiveness and competitiveness of our model. This comprehensive evaluation aims to provide insights into the model's performance across different datasets and its comparative performance against established benchmarks in the field.

Figure 14 illustrates that the YOLOv8_STN model performs similarly to state-of-the-art models. This comparison demonstrates the proposed model's effectiveness and competitiveness. Table 7 has the detailed evaluation metrics values for performance comparison.

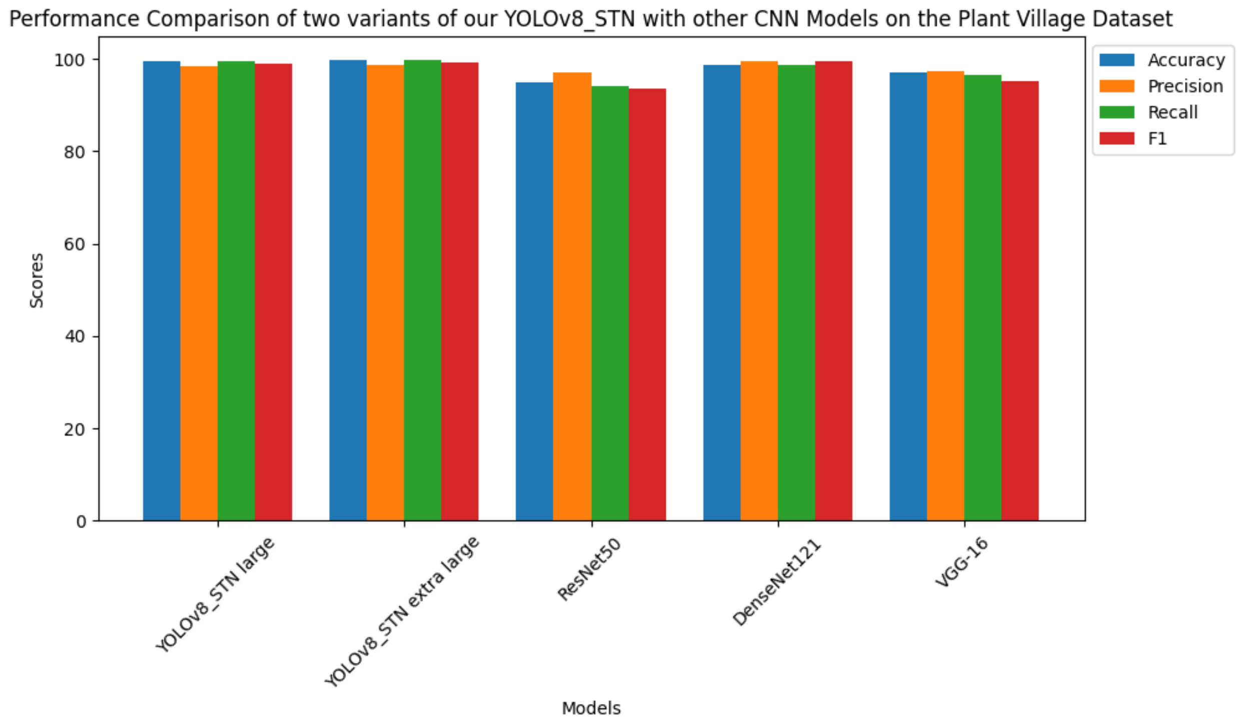


Figure 14: Performance comparison of proposed YOLOv8_STN with State-of-the-art CNNs on the Plant Village Dataset

Table 7: Evaluation Metrics of YOLOv8_STN and State-of-the-art CNNs on the Plant Village Dataset

Models	Accuracy	Precision	Recall	F1-Score
YOLOv8_STN large	99.38	98.32	99.55	98.92
YOLOv8_STN extra	99.69	98.55	99.77	99.14
ResNet50	94.75	97.13	94.14	93.62
DenseNet121	98.77	99.38	98.78	99.33
VGG-16	96.91	97.2	96.6	95.05

Chapter 5

Limitations

The study encountered several limitations that warrant discussion. First, the dataset utilised in this research exhibited poor lighting conditions and varying orientations, posing challenges to the optimal performance of state-of-the-art models such as ResNet, VGG-16, and DenseNet. Even the base paper achieved the highest accuracy of 73% on EfficientNetV2B3. The inherent complexities introduced by these environmental factors could have influenced the model's ability to generalise effectively.

Next, despite efforts to augment the dataset to balance class distributions, a notable decline in model accuracy was observed. The same observation was also evident in the base paper, where the model accuracy deteriorated after data augmentation. This discrepancy highlights the inherent difficulties in achieving balanced and representative datasets, which can significantly impact the performance of machine learning models.

Another limitation arises from the scarcity of published literature and research on uncontrolled environment datasets. The lack of comprehensive studies and benchmark datasets tailored to such real-world scenarios restricts the depth of comparative analysis and model evaluation, limiting the scope of insights gained from this research work.

Chapter 6

Conclusion And Future Scope

This research revolves around enhancing potato leaf disease detection through a Customized YOLOv8 Classifier and its variants, addressing real-world challenges such as varying environments and image quality issues. Our contributions encompassed meticulous dataset preparation, leveraging techniques like histogram equalisation, colour space conversion, and image denoising. We conducted experiments with different YOLOv8 Classifier variants, gaining valuable insights into their performance nuances. Our customised YOLOv8 Classifier, integrating a Spatial Transformer Network (STN) and additional layers, showcased improved disease spot detection accuracy.

Moreover, our study's four main components, including data preprocessing, model training, custom YOLOv8 development, and evaluation using F1-score metrics, shed light on the methodology's efficacy and limitations. Despite encountering challenges like dataset imbalances and environmental complexities, our model achieved a notable accuracy of 91% on the HSV extra-large variant of YOLOv8_STN, which is almost an 18% increment from the base paper.

There are exciting avenues for further exploration and enhancement in this domain. Expanding the range of CNN and transformer models for comparative analysis could uncover more optimal solutions. To significantly enhance the model's adaptability and generalisation capabilities, it is imperative to diligently experiment with diverse ensemble voting strategies and thoroughly evaluate its robustness on extensive datasets.

Appendices

Appendix 1 Model Training Tables

Appendix 1.1 Tables for the YOLOv8 variants

Table 8: Comparison of classification results of YOLOv8 variants on the test data without any preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	81.54	81.09	79.05	79.25
Small	86.26	87.27	83.95	84.81
Medium	87.12	87.38	83.08	84.27
Large	88.19	86.34	87.53	86.75
Extra Large	88.19	83.62	90.49	87.26

Table 9: Comparison of classification results of YOLOv8 variants on the test data with Gaussian preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	84.97	86.57	81.59	82.92
Small	87.12	88.23	85.17	85.86
Medium	86.69	86.96	82.44	83.74
Large	89.05	89.37	86.1	87.11
Extra Large	89.69	87.3	90.77	88.63

Table 10: Comparison of classification results of YOLOv8 variants on the test data with CLAHE preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	84.54	81.02	84.11	82.02
Small	87.55	84.92	88.18	85.85
Medium	88.19	85.23	87.63	86.06
Large	89.27	85.94	87.02	86.41
Extra Large	89.69	89.45	88.86	88.85

Table 11: Comparison of classification results of YOLOv8 variants on the test data with Median preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	83.47	82.11	82.08	81.92
Small	88.41	90.44	87.5	88.76
Medium	87.55	86.06	86.07	86
Large	88.41	87.1	89.51	88.13
Extra Large	89.27	87.24	88.98	88.03

Table 12: Comparison of classification results of YOLOv8 variants on the test data with HSV preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	86.05	84.35	86.92	84.89
Small	87.76	85.3	88.68	86.6
Medium	88.62	87.28	88.26	87.71
Large	88.41	88.36	85.09	86.47
Extra Large	90.55	88.92	88.68	88.77

Table 13: Comparison of classification results of YOLOv8 variants on the test data with HE preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	86.73	86.47	84.34	85.2
Small	87.76	85.69	89.41	87.25
Medium	88.62	86.44	89.16	87.33
Large	88.06	85.21	88.4	86.53
Extra Large	88.41	85.28	88.85	86.8

Appendix 1.2 Tables for the YOLOv8_STN variants

Table 14: Comparison of classification results of YOLOv8 variants on the test data without any preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	86.26	84.56	88.27	86.01
Small	88.84	87.39	89.62	88.31
Medium	87.55	84.7	86.57	85.31
Large	89.69	88.85	88.77	88.71
Extra Large	90.12	87.88	88.56	87.82

Table 15: Comparison of classification results of YOLOv8 variants on the test data with Gaussian preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	84.76	82.14	84.16	82.77
Small	89.91	89.01	90.46	89.52
Medium	87.98	85.07	87.64	85.55
Large	89.48	88.82	86.92	87.75
Extra Large	90.77	88.61	88.08	88.21

Table 16: Comparison of classification results of YOLOv8 variants on the test data with CLAHE preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	87.33	84.25	85.12	84.55
Small	87.76	87.95	86.28	86.99
Medium	89.05	85.68	85.91	85.75
Large	89.27	86.82	87.9	87.27
Extra Large	90.27	90.01	88.26	89.12

Table 17: Comparison of classification results of YOLOv8 variants on the test data with Median preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	84.97	82.82	84.52	83.3
Small	87.33	85.89	87.07	86.36
Medium	87.98	86.5	84.21	85.23
Large	89.27	88.56	90.56	89.34
Extra Large	89.69	88.72	90.08	88.66

Table 18: Comparison of classification results of YOLOv8 variants on the test data with HSV preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	87.98	86.23	85.46	85.62
Small	88.72	86.25	84.33	85.08
Medium	89.06	85.87	86.38	86.05
Large	89.69	87.9	87.17	87.39
Extra Large	90.81	92.97	87.93	90.02

Table 19: Comparison of classification results of YOLOv8 variants on the test data with HE preprocessing

Variant	Accuracy	Precision	Recall	F1 Score
Nano	85.68	84.62	82.94	83.44
Small	88.19	88.96	87.43	88.14
Medium	87.55	89.41	84.78	86.59
Large	89.05	89.37	86.5	87.55
Extra Large	89.15	89.16	89.14	88.96

Appendix 2 Code Snippets

Appendix 2.1 CLAHE Preprocessing Code

```
import os
import cv2

# Define the root directory of your dataset
root_dir = "Potato Leaf Disease Dataset in Uncontrolled Environment"

# Define the output directory for storing the processed images
output_dir = "processedDataset"

# Create the output directory if it doesn't exist
if not os.path.exists(output_dir):
    os.makedirs(output_dir)

# Function to perform CLAHE on an image
def perform_clahe(image):
    lab = cv2.cvtColor(image, cv2.CLRBGR2LAB)
    l, a, b = cv2.split(lab)
    clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(8, 8))
    cl = clahe.apply(l)
    enhanced_lab = cv2.merge((cl, a, b))
    enhanced_image = cv2.cvtColor(enhanced_lab, cv2.CLRBGR2LAB)
    return enhanced_image
```

```
import cv2
from google.colab.patches import cv2_imshow

def show_images_side_by_side(image_path1, image_path2):
    # Read images
    image1 = cv2.imread(image_path1)
    image2 = cv2.imread(image_path2)

    # Check if images are loaded successfully
    if image1 is None or image2 is None:
        print("Error: One or both images could not be loaded.")
        return

    # Resize images to the same height (optional)
    max_height = max(image1.shape[0], image2.shape[0])
    image1 = cv2.resize(image1, (int(image1.shape[1] * max_height / image1.shape[0]),
max_height))
    image2 = cv2.resize(image2, (int(image2.shape[1] * max_height / image2.shape[0]),
max_height))

    # Add captions to images
    cv2.putText(image1, "Normal Image", (20, 40), cv2.FONT_HERSHEY_SIMPLEX, 1,
(255, 255, 255), 2)
    cv2.putText(image2, "CLAHE Image", (20, 40), cv2.FONT_HERSHEY_SIMPLEX,
1, (255, 255, 255), 2)

    # Concatenate images horizontally
    combined_image = cv2.hconcat([image1, image2])

    # Display the combined image
    cv2_imshow(combined_image)
```

```
# Example usage
image_path1 = "/content/Potato Leaf Disease Dataset in Uncontrolled
Environment/Fungi/1692332350534.jpg"
image_path2 = "/content/processedDataset/Fungi/1692332350534.jpg"
show_images_side_by_side(image_path1, image_path2)
```

Appendix 2.2 HSV Preprocessing Code

```
import os
import cv2

# Define the root directory of your dataset
root_dir = "Potato Leaf Disease Dataset in Uncontrolled Environment"

# Define the output directory for storing the processed images
output_dir = "processedDataset"
def perform_hsv_manipulation(image):
    # Convert the image to HSV color space
    hsv = cv2.cvtColor(image, cv2.CLRBGR2HSV)

    # Split the HSV image into components
    h, s, v = cv2.split(hsv)

    # Example manipulation: Increase the saturation (S) component
    s = cv2.add(s, 30)  # Increase saturation by 30 (adjust as needed)

    # Merge the manipulated components back into the HSV image
    manipulated_hsv = cv2.merge([h, s, v])
    manipulated_image = cv2.cvtColor(manipulated_hsv, cv2.CLRHSV2BGR)

    return manipulated_image

import cv2
def show_images_side_by_side(image_path1, image_path2):
    image1 = cv2.imread(image_path1)
    image2 = cv2.imread(image_path2)
    image1 = cv2.resize(image1, (int(image1.shape[1] * max_height / image1.shape[0]),
max_height))
    image2 = cv2.resize(image2, (int(image2.shape[1] * max_height / image2.shape[0]),
max_height))

    # Add captions to images
    cv2.putText(image1, "Normal Image", (20, 40), cv2.FONT_HERSHEY_SIMPLEX, 1,
(255, 255, 255), 2)
    cv2.putText(image2, "HSV Image", (20, 40), cv2.FONT_HERSHEY_SIMPLEX, 1,
(255, 255, 255), 2)

    # Concatenate images horizontally
    combined_image = cv2.hconcat([image1, image2])

    # Display the combined image
    cv2_imshow(combined_image)

# Example usage
image_path1 = "/content/Potato Leaf Disease Dataset in Uncontrolled
Environment/Fungi/1692332350534.jpg"
image_path2 = "/content/processedDataset/Fungi/1692332350534.jpg"
show_images_side_by_side(image_path1, image_path2)
```

Appendix 2.3 HE Preprocessing Code

```
import os
import cv2
import numpy as np

# Define the root directory of your dataset
root_dir = "Potato Leaf Disease Dataset in Uncontrolled Environment"

output_dir = "processedDataset"

def enhance_disease_spots(image):
    lab = cv2.cvtColor(image, cv2.CLRBGR2LAB)
    l, a, b = cv2.split(lab)
    l_enhanced = cv2.equalizeHist(l)
    enhanced_lab = cv2.merge([l_enhanced, a, b])
    enhanced_image = cv2.cvtColor(enhanced_lab, cv2.COLOR_LAB2BGR)
    return enhanced_image

import cv2
from google.colab.patches import cv2_imshow

def show_images_side_by_side(image_path1, image_path2):
    # Read images
    image1 = cv2.imread(image_path1)
    image2 = cv2.imread(image_path2)
    image1 = cv2.resize(image1, (int(image1.shape[1] * max_height / image1.shape[0]),
max_height))
    image2 = cv2.resize(image2, (int(image2.shape[1] * max_height / image2.shape[0]),
max_height))

    # Add captions to images
    cv2.putText(image1, "Normal Image", (20, 40), cv2.FONTHERSHEYSIMPLEX, 1,
(255, 255, 255), 2)
    cv2.putText(image2, "HE Image", (20, 40), cv2.FONTHERSHEYSIMPLEX, 1, (255,
255, 255), 2)

    # Concatenate images horizontally
    combined_image = cv2.hconcat([image1, image2])

    # Display the combined image
    cv2_imshow(combined_image)

# Example usage
image_path1 = "/content/Potato Leaf Disease Dataset in Uncontrolled
Environment/Fungi/1692332350534.jpg"
image_path2 = "/content/processedDataset/Fungi/1692332350534.jpg"
show_images_side_by_side(image_path1, image_path2)
```

Appendix 2.4 Median Filtering Code

```
import cv2
import os

# Function to apply median filtering on an image
def apply_median_filter(image_path, output_path):
    img = cv2.imread(image_path)
    median = cv2.medianBlur(img, 5)  # Adjust the kernel size as needed
    cv2.imwrite(output_path, median)

# Input and Output directories
input_folder = "/content/Potato Leaf Disease Dataset in Uncontrolled Environment"
output_folder = "/content/24jan-potato-median-filtered"

# Create output folder if not exists
os.makedirs(output_folder, exist_ok=True)

# Loop through each class folder in the input dataset
for class_folder in os.listdir(input_folder):
    class_input_folder = os.path.join(input_folder, class_folder)
    class_output_folder = os.path.join(output_folder, class_folder)

    # Create output class folder if not exists
    os.makedirs(class_output_folder, exist_ok=True)

    # Loop through each image in the class folder
    for root, dirs, files in os.walk(class_input_folder):
        for file in files:
            # Construct the full path of the input and output images
            input_image_path = os.path.join(root, file)
            output_image_path = os.path.join(class_output_folder, file)

            # Apply median filtering and save the result
            apply_median_filter(input_image_path, output_image_path)

print("Median filtering and feature extraction complete.")
```

```
import cv2
from google.colab.patches import cv2_imshow

def show_images_side_by_side(image_path1, image_path2):
    # Read images
    image1 = cv2.imread(image_path1)
    image2 = cv2.imread(image_path2)

    # Check if images are loaded successfully
    if image1 is None or image2 is None:
        print("Error: One or both images could not be loaded.")
        return

    # Resize images to the same height (optional)
    max_height = max(image1.shape[0], image2.shape[0])
    image1 = cv2.resize(image1, (int(image1.shape[1] * max_height / image1.shape[0]),
max_height))
    image2 = cv2.resize(image2, (int(image2.shape[1] * max_height / image2.shape[0]),
max_height))

    # Add captions to images
    cv2.putText(image1, "Non-filtered Image", (20, 40), cv2.FONTHERSHEYSIMPLEX,
1, (255, 255, 255), 2)
    cv2.putText(image2, "Median Filtered Image", (20, 40),
cv2.FONTHERSHEYSIMPLEX, 1, (255, 255, 255), 2)

    # Concatenate images horizontally
    combined_image = cv2.hconcat([image1, image2])

    # Display the combined image
    cv2_imshow(combined_image)
```

```
# Example usage
image_path1 = "/content/Potato Leaf Disease Dataset in Uncontrolled
Environment/Fungi/1692332350534.jpg"
image_path2 = "/content/24jan-potato-median-filtered/Fungi/1692332350534.jpg"
show_images_side_by_side(image_path1, image_path2)
```

Appendix 2.5 Gaussian Filtering Code

```
import os
import cv2

def apply_gaussian_filter(image, kernel_size=(5, 5), sigmaX=0):
    # Apply Gaussian blur
    filtered_image = cv2.GaussianBlur(image, kernel_size, sigmaX)
    return filtered_image



def process_dataset(input_dataset_folder, output_dataset_folder):
    # Process each subfolder in the dataset
    for folder_name in os.listdir(input_dataset_folder):
        input_folder = os.path.join(input_dataset_folder, folder_name)
        output_folder = os.path.join(output_dataset_folder, folder_name)
        process_images_in_folder(input_folder, output_folder)

if __name__ == "__main__":
    input_dataset_folder = "Potato Leaf Disease Dataset in Uncontrolled Environment"
    output_dataset_folder = "7feb-new-potato-gaussian-filtered"
    process_dataset(input_dataset_folder, output_dataset_folder)
```

REFERENCES

[1] India: production volume of potato 2023 | Statista. (n.d.). Statista.

[2] Shabrina, N. H., Indarti, S., Maharani, R., Kristiyanti, D. A., Irmawati, Prastomo, N., & Adilah M, T. (2024, February). A novel dataset of potato leaf disease in an uncontrolled environment. Data in Brief, 52, 109955.

[3] Wang, G., Chen, Y., An, P., Hu, H., Hu, J., & Huang, T. (2023, August 15). UAV-YOLOv8: A Small-Object-Detection Model Based on Improved YOLOv8 for UAV Aerial Photography Scenarios. Sensors. https://doi.org/10.3390/s23167190

[4] Wu, T., & Dong, Y. (2023, December 5). YOLO-SE: Improved YOLOv8 for Remote Sensing Object Detection and Recognition. Applied Sciences. https://doi.org/10.3390/app132412977

[5] Sary, I. P., Andromeda, S., & Armin, E. U. (2023, June 30). Performance Comparison of YOLOv5 and YOLOv8 Architectures in Human Detection using Aerial Images. Ultima Computing: Jurnal Sistem Komputer, 8–13. https://doi.org/10.31937/sk.v15i1.3204

[6] Al-Adhaileh, M. H., Verma, A., Aldhyani, T. H. H., & Koundal, D. (2023, March 21). Potato Blight Detection Using Fine-Tuned CNN Architecture. Mathematics. https://doi.org/10.3390/math11061516

[7] Arshad, F., Mateen, M., Hayat, S., Wardah, M., Al-Huda, Z., Gu, Y. H., & Al-Antari, M. A. (2023, September 1). PLDPNet: End-to-end hybrid deep learning framework for potato leaf disease prediction. Alexandria Engineering Journal. https://doi.org/10.1016/j.aej.2023.07.076

[8] Jung, H. K., & Choi, G. S. (2022, July 19). Improved YOLOv5: Efficient Object Detection Using Drone Images under Various Conditions. Applied Sciences. https://doi.org/10.3390/app12147255

[9] Rashid, J., Khan, I., Ali, G., Almotiri, S. H., AlGhamdi, M. A., & Masood, K. (2021, August 26). Multi-Level Deep Learning Model for Potato Leaf Disease Recognition. MDPI.

[10] A Convolutional Neural Network Based Potato Leaf Diseases Detection Using Sequential Model. (2023, January 24). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10080063/

[11] Detection of Potato Disease Using Image Segmentation and Machine Learning. (2020, August 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9198563

[12] Agarwal, M., Singh, A., Arjaria, S., Sinha, A., & Gupta, S. (2020). ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network. Procedia Computer Science, 167, 293–301. https://doi.org/10.1016/j.procs.2020.03.225

[13] Shah, D. I., Trivedi, V., Sheth, V., Shah, A., & Chauhan, U. (2022, June 1). ResTS: Residual Deep interpretable architecture for plant disease detection. Information Processing in Agriculture. https://doi.org/10.1016/j.inpa.2021.06.001

[14] Tambe, U. Y., Shobanadevi, A., Shanthini, A., & Hsu, H. C. (2023, November 4). Potato Leaf Disease Classification using Deep Learning: A Convolutional Neural Network Approach. arXiv.org. https://arxiv.org/abs/2311.02338

[15] Early Detection of Potato Leaf Diseases using Convolutional Neural Network with Web Application. (2022, June 17). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/9848975