Journal of Computer Science

Research Article

Implementation and Performance Evaluation of YOLOv8 for Wheat Fungal Disease Detection

Shivani Sood[1], Shallu Duggal[1], Monika Sethi[2], Chander Prabha[2], Prakash Srivastava[3] Mohammad Zubair Khan[4], Amna Bamaqa[5] and Abdulaziz Alblwi[5]

[1]*School of Computer Applications, Lovely Professional University, Phagwara, Punjab, India*
[2]*Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, 140401, India*
[3]*Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun, India*
[4]*Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah 42351, Saudi Arabia*
[5]*Department of Computer Science and Information, Applied College, Taibah University, Medinah 41461, Saudi Arabia*

Article history
Received: 04-01-2025
Revised: 02-08-2025
Accepted: 13-02-2026

Corresponding Author:
Chander Prabha
Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, 140401, India
Email: prabhanice@gmail.com

Abstract: Among the many difficulties in producing wheat, a vital food source for the world, there are biotic variables like fungal diseases, which drastically lower crop output. Around the world, biotrophic fungi particularly, leaf rust, powdery mildew, and yellow rusts-have become a constant threat to wheat production. Traditional methods of identification and mitigation are labor-intensive, slow, and imprecise, limiting therefore early detection and intervention of wheat diseases is essential. Convolutional Neural Networks (CNNs), are the recent development in deep learning techniques, have shown promising disease detection solutions. Among these, the YOLO (You Only Look Once) models-particularly YOLOv8-have shown remarkable efficiency in identifying and categorizing wheat diseases like leaf rust, yellow rust, and powdery mildew. In this study, the YOLOv8 model was trained on a wheat image dataset, and achieved a mean average precision (mAP) of 0.99. The outcomes demonstrate the model's reliability in identifying diseases under a range of circumstances, emphasizing the significance of maximizing training time to prevent overfitting. To further improve model performance, future studies should investigate data augmentation, hyperparameter adjustment, and real-time deployment in agricultural settings. The study demonstrates how deep learning models can enhance crop monitoring, disease control, and eventually agricultural output.

Keywords: Yolov8, Fungal Disease Detection, Wheat Crop Diseases, Precision Agriculture, Automatic Disease Recognition

Introduction

A lack of technical knowledge leads to the destruction of many crops. One of the important sources of income for people in India is agriculture. Farmers grow a variety of crops, but diseases often destroy them due to lack of knowledge. Plant disease is the primary cause of crop damage in India. Different plants suffer from different diseases. Experts in agriculture inspect the plant leaf in the center to determine the disease's cause. However, this method of plant disease detection was expensive and time-consuming. Therefore, a more advanced technique will be needed to identify leaf diseases. Wheat is one of the most important and largest food resources in the world; its production needs to be continuously improved to fulfill the needs of the growing human population. After rice, wheat is the second most consumed grain worldwide and a significant cereal crop. For more than one-third of the world's population, it is a staple food and most consumed by people in their daily life. Numerous biotic and abiotic factors, including soil, temperature, pests, bacteria, and other environmental conditions, can affect wheat production (Li et al., 2023; Sereda et al., 2023). Specialist diseases known as biotrophic fungi, such as rusts, powdery mildews, and smuts, significantly impact agriculture. Biotrophic fungi infect the living plant cells without causing abrupt cell death. These fungi are from various lineages, such as rusts (Pucciniales) (Luo et al.,

2024). Approximately 8,000 species of rust fungus, which are among the most varied fungal diseases, infected from various host plants (Lorrain et al., 2019). Even though scientists have tryied to develop wheat plants that can resist diseases, fungi like Puccinia graminis f. sp. tritici have found ways to overcome these defenses. Additionally, these diseases have a major effect on wheat quality and productivity. The majority of wheat diseases affect the leaves; the most prevalent ones are powdery mildew, leaf rust, and yellow rust. These are all fungal infections that can significantly reduce wheat yields (Kartikeyan and Shrivastava, 2021; Khan et al., 2022). Understanding wheat's evolution and genetic diversity across its species and their relatives are crucial for improving wheat crop production (Peng et al., 2011). Precisely, identifying growing regions of wheat is essential for efficient crop monitoring, disease control, and yield estimation, as it is a globally important food crop (Bond and Liefert, 2016). However, timely prevention and treatment of wheat diseases is not possible due to farmers' lack of professional knowledge, small size, and difficulty in observing early-stage infections. Therefore, one of the crucial components for the advancement of modern agriculture is the ability to accurately identify wheat diseases which can efficiently provide crop production.

These days, science and technology are expanding quickly thanks to numerous groundbreaking discoveries that simplify and ease people's lives. The agriculture industry now has more options for addressing these issues because of the contemporary use of technologies like machine learning and deep learning. Deep learning technology is one of the most common ways to identify and classify plant diseases. With an increasing improvement in the accuracy of detection rates, an increasing number of research studies in recent years have concentrated on crop management and disease detection using these technologies (Sood and Singh, 2020; 2021; 2022; 2024; Sood et al., 2021; 2022; Wijayanto et al., 2023). Deep learning is a branch of machine learning that attempts to imitate the way the human brain processes information and makes decisions. CNNs are a key component of deep learning in many domains, including genomics, image and speech recognition, natural language processing, and speech recognition (Wijayanto et al., 2023). Its usefulness encompasses a wide range of actual applications, including banking, healthcare, and even education, as demonstrated by the CNN and e-learning modeling techniques presented in Alahmari et al. (2023). Significantly, research using deep learning to detect diseases in certain plant species has produced promising results. Many important studies in this field have investigated plant disease detection. For example, Uoc et al. (2022) used the YOLOv4 network for processing images of leaf diseases to detect diseases on cucumber plants. The authors utilized a dataset with more than 7,000 images and achieved remarkable accuracy rates that exceeded 80%. However, the accuracy of this study was limited because it only used a CNN model. Deep learning techniques for tracking and plant leaf disease detection were examined in another work (Saleem et al., 2019). It looked at imagery at different infection levels and developed a model to evaluate how the disease progresses during the life cycle of the plant. This study also looked into other variables, like illumination, learning rate, and dataset size, that affect image quality. Researchers used YOLOv3 in different settings to identify leaf blast and brown spot diseases. However, this approach's limited versatility and restricted usage of images with white backgrounds made it less practical. (Agbulos et al., 2021).

The detection and categorization of leaf diseases are significantly aided by computers and software. Numerous image processing and pattern recognition algorithms are available for use in leaf disease identification. Early disease detection is critical to prevent agricultural losses. Each disease should have a treatment, which should be entered into the database to stop damage from happening later (Ampatzidis et al., 2017). As a result, it is imperative to find a quick and accurate method for the treatment of plant diseases to maintain the balance between agriculture and the environment. During rural harvests, leaves play a vital role in providing information on the quantity and composition of agricultural produce. A few factors, including weed growth, soil infertility, and environmental change, affect the production of food. In addition, plant or leaf diseases are a global threat to the growth of horticultural products and a cause of financial hardship (Duro et al., 2012). The identification and classification of target fruits has been the subject of extensive study and applied efforts for the last decade or more. Traditional methods examine individual characteristics, including color, texture, and geometric shape. In Si et al. (2015), the author used color variation analysis to distinguish apples from background images. They used the multi-feature fusion geometric form and color features analysis, which uses color, edge, orientation, and intensity characteristics combined together.

YOLOv5 and YOLOv8 are two distinct versions of the advanced deep-learning model that are capable of properly identifying diseases and evaluating the general health of plant leaves. These models identify crop diseases more accurately with less time and effort, ultimately increasing yields and revenues for farmers and the stakeholders (Orchi et al., 2023; Slimani et al., 2023; Wang et al., 2024). In this paper, we have detected the wheat plant diseases using YOLOv8, the most prominently used method of object detection nowadays. Furthermore, we have divided the paper into distinct sections.

Literature Review

In this section, we have performed the literature review and analyzed how the modern deep learning-based object detection methods are utilized for detecting plant diseases.

In Venkataramanan et al. (2019), the authors introduce a deep learning approach for plant disease detection by analyzing leaf images. A YOLOv3 detector isolates the leaf, and ResNet18 models classify the plant type and potential diseases. Using transfer learning, the model fine-tunes the last layers while freezing the initial ones, achieving 78.00% accuracy after 25 epochs, which was further improved to 96.00% by rotating input images for better predictions.

To identify healthy tomato leaves and four diseases, powdery mildew, blight, leaf mold fungus, and ToMV, Zhang et al. (2020) used a Faster RCNN model. To increase feature extraction, the authors substituted the depth residual network for the VGG16 technique. They also provided a k-means clustering algorithm to enhance anchor boxes according to the clustering outcomes. Compared to the originalr R-CNN, the revised approach delivers faster detection speed and higher recognition accuracy by 2.71%.

Anjanadevi et al. (2020) has implemented CNNs for object classification and achieved a training accuracy of 85.30%. This model has limitations such as prolonged training times and less accuracy when evaluated with images that differ from the training set. So, another model called 'Improved-Detect' was suggested, which uses the YOLO architecture and adds features like residual blocks and skip connections, while also using the Darknet-53 network to better capture important details. This model achieved an impressive success rate of 99.10% in classifying plant diseases.

In Morbekar et al. (2020), the authors proposed the YOLO technique for plant disease detection. The system uses the PlantVillage dataset and applies changes to individual pixels along with Generative Adversarial Networks (GANs) to create more data, helping the model perform better. The architecture of YOLO effectively predicts bounding boxes and class probabilities, which improves disease detection speed and reports an accuracy of 100%.

Using an enhanced YOLOv4 method, Roy and Bhaduri (2021) proposed a deep learning-based object recognition model for multi-class plant diseases, with particular focus on apple plant diseases. The model outperforms prior models with a precision improvement of 9.05% and an F1-score increase of 7.60%, achieving a mean average precision (mAP) of 91.20% and an F1-score of 95.90% at a detection rate of 56.90 frames per second (FPS). Their proposed framework effectively detects diseases in complex orchard settings by managing problems like background noise and recognizing details at different scales, showing potential for broader applications in automated farming.

With regard to plant disease identification, Roy et al. (2022), suggested a high-performance real-time object detection system that tackles issues including multi-scale object classes, irregular morphology, and dense distribution. Using DenseNet to better transfer features, adding extra residual blocks to enhance feature extraction, and incorporating Spatial Pyramid Pooling (SPP) to widen the receptive field, the model is an upgraded version of YOLOv4. By enhancing the extraction of non-linear features, the Hard-Swish activation function increases accuracy. The model was faster, used 70.19 FPS, and proved more accurate than current detection techniques. Their model achieved an mAP of 96.29% and precision of 90.33% when tested on four tomato plant diseases.

In the study by Eunice et al. (2022), the authors refined with pre-trained models (DenseNet-121, ResNet50, VGG-16, and Inception V4) on the PlantVillage dataset using CNN-based transfer learning approaches. DenseNet121 attained a classification accuracy of 99.81%. The results indicate that DenseNet-121 performs better than other models in terms of sensitivity, specificity, and F1- score, among other evaluation criteria.

Kaur et al. (2022) identified tomato leaf infections employing a dataset of 1,610 images to propose and modify the Mask Region Convolutional Neural Network (Mask R-CNN) for the automated segmentation and detection of tomato plant leaf diseases. With a mean average precision (mAP) of 0.88, an F1-score of 0.912, and an accuracy of 0.98, the model outperforms existing models in terms of metrics and detects events much faster.

Kurmi et al. (2022) proposed a multi-stage technique for plant disease classification. Firstly, foreground leaf regions were separated from backgrounds before feature extraction using color transformation. After that, the features are extracted using the Adaptive Analytic Wavelet Transform (AAWT). The classification combines Bag of Visual Words (BoW), Fisher Vectors (FV), and AAWT features. For classification, they used SVM, MLP, and logistic regression algorithms that were trained and tested on the PlantVillage dataset and achieved an accuracy of 94.07% and an AUC of 0.96.

Lee et al. (2023) presented a novel approach to crop disease diagnosis by combining object detection and image captioning, two cutting-edge deep learning techniques. With an average Bilingual Evaluationy (BLEU) score of 64.96%. They used image captioning models to demonstrate their capacity to produce semantically coherent and grammatically accurate diagnostic words that categorize disease symptoms as per the severity levels. The YOLOv5 model for object detection was employed; it showed promising results with improvement of a mean average precision with a 50% threshold value.

Table 1 illustrates a literature review wherein we have described and summarized the other researchers' recent work for detecting the plant diseases. After reviewing the literature, it is determined that most of the authors used larger amount of dataset to train various deep learning models such as YOLOv3, YOLOv4, Faster R-CNN, etc. and achieved the highest value of mAP, and accuracy. However, in this manuscript, we have trained the state-of-the-art YOLOv8 and performed the training on smaller dataset.

Dataset Description

In this paper, we have used the WFD2020 wheat disease dataset, which contains different fungal diseases. Currently, various object detection models require images and their annotations for training. For every object detection model, there is a separate annotation format. For example: Faster R-CNN based models take the *.xml file format; however, the YOLOv8 model used the *.txt format during the training process. Since YOLOv8 requires annotations in .txt format, this study employed the model accordingly. In the current study, we used an AI-based image annotation tool called Computer Vision Annotation Tool (CVAT) to label and annotate the images. It is an open-source tool developed by Intel that is used to manually annotate image and video datasets, especially for machine learning and computer vision-based algorithms.

Additionally, many individuals are currently using these tools for image classification and to address segmentation-related problems. Using this tool, different types of annotation can be created, such as polygons, bounding boxes, polylines, etc. Fig. 1 displays the sampled images of wheat fungal diseases that are being used for experimentation. In Fig. 1:

a) Represents the yellow rust, where the rust pustules look like pores that are yellow in circular shapes
b) Shows the sample of powdery mildew of white color
c) Shows a Leaf rust sample that is brown in an oval shape

Table 1: Summary of Tomato Disease Detection Improvement

References	Methodology	Dataset	Dataset Size	Image Size	Results
Venkataram et al. (2019)	YOLOv3	PlantVillage	36,148 images	256×256	96% Accuracy Achieved
Roy and Bhaduri (2021)	YOLOv4	PlantVillage	2,000 images	512×512	mAP = 91.20%, F1-Score = 95.90%
Roy et al. (2022)	Improved version of YOLOv4	PlantVillage	12,000 images	416×416	mAP = 96.29%
Eunice et al. (2022)	Convolutional Neural Networks (CNN) and transfer learning approaches: DenseNet-121, ResNet-50, VGG-16, and Inception V4	PlantVillage	54,305 images	299×299	DenseNet-121 attained a classification accuracy of 99.81%
Kaur et al. (2022)	Mask R-CNN	Primary dataset	1,610 images	1024×1024	mAP = 88%, F1score = 92%, and accuracy = 98%
Zhang et al. (2020)	Customised Faster RCNN, K-means clustering	Dataset from AIChallenger having 61 categories	4,178 images	150×150	RCNN-res101 algorithm achieved map 98.54%
Morbekar et al. (2020)	YOLOv3 based object detection algorithm	PlantVillage dataset of 54,306 images, 25 classes of healthy and diseased plant species from Kaggle	54,306 images	256×	mAP approaching 100%
Anjanadevi et al. (2020)	MobileNet, Inception v2 and ResNet -101 for Feature Extraction, DarkNet-53 for Classification	Plantdoc and Plant Village datasets	54,306 images	416×416	mAP achieved 99.10%
Lee et al. (2023)	Features extracted from InceptionV3, YOLOv5m variants for detection	Dataset of 9 crop diseases from AI hub Korean platform	1,23,913 images	299×299	Accuracy of 64.96%
Kurmi et al. (2022)	AAWT technique for feature extraction and decomposition and SVM, MLP for classification	PlantVillage dataset consisting of 14 various crops	54,309 images	256×256	SVM achieved accuracy of 94%

(a) Yellow Rust (b) Powdery Mildew (b) Leaf Rust

Fig. 1: Sample of diseased images

Table 2 describes the class-wise description of the dataset, which contains the images and generated its instances for training the model. A total of 63 images of wheat fungal diseases of three classes, powdery mildew, yellow rust, and leaf rust, containing 24, 23, and 16 images, respectively. These diseases commonly appear during the growing stage of the wheat crop. After annotation, the region of interest is generated for each instance having values 116, 49, and 25, respectively.

Table 2: Class-wise Description of the dataset with images and its instances

Classes	Total Images	Instances
Powdery Mildew	24	116
Yellow Rust	23	49
Leaf Rust	16	25
Total	63	190

Methodology

In the current study, we have used YOLOv8 and Faster R-CNN one-stage and two-stage detector model, respectively, for detecting the wheat plant diseases. As there are various algorithms available that can be used for detecting plant diseases. However, these models mostly used by several domains due to its speed, real-time processing capacity, and its architectural efficiency. A comparison of the YOLOv8 model with other popular two-stage detector model Faster R-CNN is given below: Inference Speed and Real-Time Performance: The single-stage detection architecture of YOLOv8 aims for real-time object identification that enables faster inference times. YOLOv8 is particularly effective for tasks requiring quick responses, such as industrial inspection, autonomous vehicles, and surveillance, as detection takes place in a single network trip. On the other hand, Faster R-CNN is a two-stage detector; that generates region proposals in the first stage and then classifies, these regions in the second stage.

Architectural Simplicity and Efficiency: YOLOv8 simplifies object detection that enables faster recognition with fewer processing resources by directly predicting bounding boxes and class probabilities from images in a single feed-forward step. Faster R-CNN uses a more complex pipeline, comprising separate networks for bounding box refinement and classification with a Region Proposal Network (RPN) to generate regions of interest.

This complexity raises by processing costs and memory needs, which may be too much for systems with limited resources.

Accuracy and Performance Trade-off: YOLOv8 carefully balances speed and precision. YOLOv8 improves upon previous versions of the algorithm in several ways, bringing it closer to Faster R-CNN accuracy. These include handling small objects better and more effective anchor-free recognition. YOLOv8 models were known for prioritizing speed above than the accuracy in the past. Faster R-CNNs are generally linked to higher accuracy, especially for precision-demanding tasks such as fine-grained detection and complex object applications. However, the accuracy gap between the two models has narrowed, a big thanks to YOLOv8's improvements, YOLOv8 competitive in terms of both accuracy and speed.

Flexibility and Adaptability: YOLOv8 model has end-to-end pipeline which adapts well to various object detection tasks and is easily customizable for unique applications. Developing faster R-CNN requires more engineering and optimization, particularly when deploying it in new environments or on constrained hardware. Because of its higher computing requirement, it is less suitable for environments with limited processing capabilities.

Hardware Requirements: YOLOv8 is more suited for edge computing because of its lighter weight and higher

efficiency. It can also run smoothly on lightweight devices, such as embedded systems or mobile devices. Faster RCNN requires more computational resources, such as more memory and processing power; hence, it is more suited for desktop environments and high-end GPUs than edge devices.

Fig. 2 shows the architecture of YOLOv8, which consists of four main blocks: Backbone, neck, head, and post-processing blocks.

Backbone: The backbone extracts features at various scales from the input image. This is essential for identifying objects of different sizes and gathering hierarchical data, such as object structures, edges, and textures. Typically, YOLOv8 uses a Convolutional Neural Network (CNN) as its backbone, often combining it with Cross-Stage Partial Networks (CSPNet). CSPNet divides the feature map into two sections, one bypassing layers and the other passing through them. These sections are then combined by preserving the gradient flow and lowering computational cost, the network becomes more accurate and faster. We frequently use several convolutional layers with batch normalization and activation functions like SiLU or Leaky ReLU. The backbone slowly reduces the size of the image while making the feature maps more detailed by using methods like max-pooling and stride convolution. Its main function is feature extraction, which turns the unprocessed input image into feature representations that encode visual data at various abstraction levels. The neck receives these feature maps and performs additional processing.

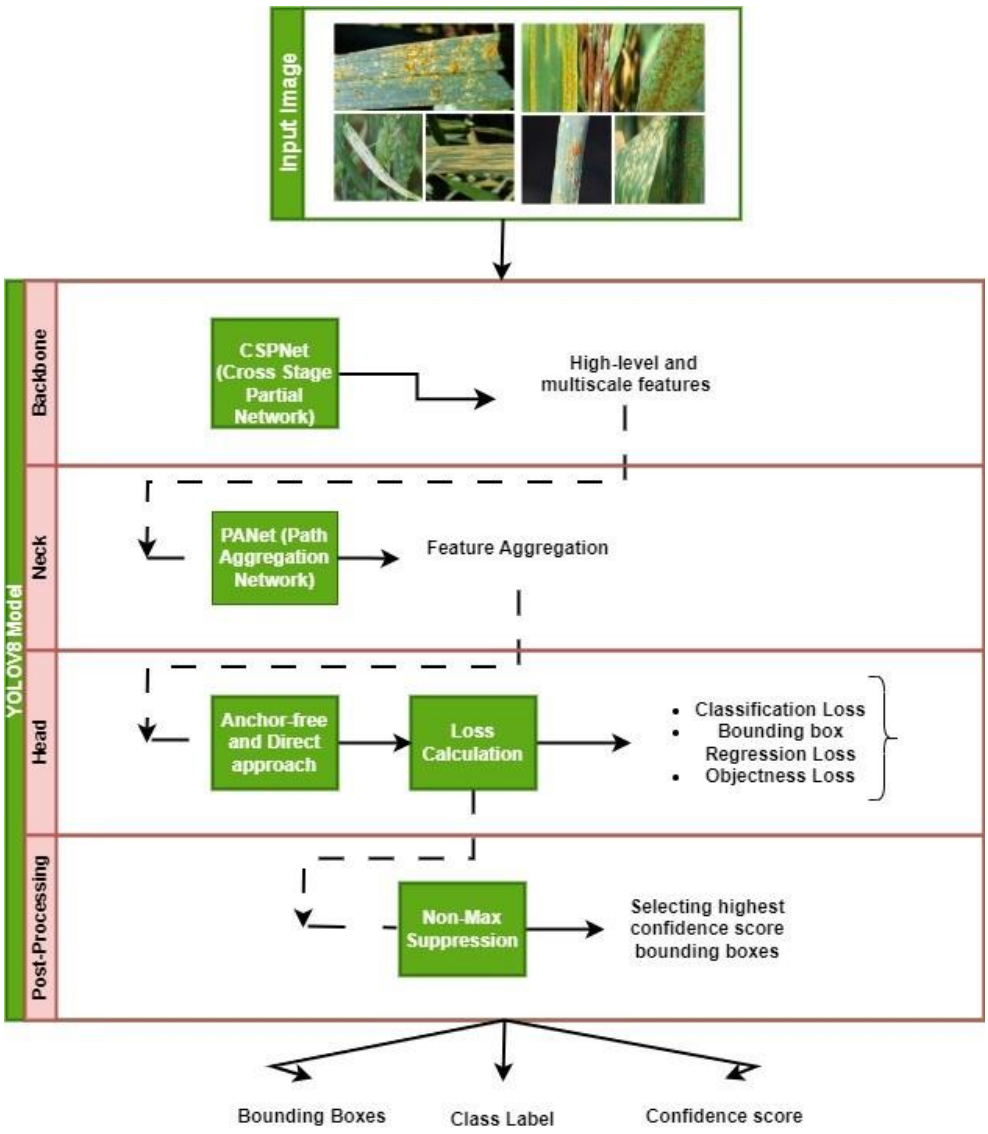


Fig. 2: YOLOv8 Architecture

Neck: Through the integration of data from various scales, the neck component enhances the feature maps generated by the Backbone. This is essential for identifying items of different sizes, particularly tiny things in complex backgrounds. YOLOv8 usually combines the Path Aggregation Network (PAN) and Feature Pyramid Network (FPN):

i) FPN: By integrating high-resolution, low-semantic characteristics with low-resolution, high-semantic information, FPN improves the network's object detection capabilities. High-level feature maps are upsampled and concatenated with lower-level feature maps using a top-down architecture
ii) PAN: To further improve feature fusion, PAN adds a bottom-up approach. This method improves localization by more effectively integrating features from several layers

The network can recognize both small and large objects thanks to the Neck's improved feature representation robustness. It makes easier for data to move between layers, which helps the network create fine-grained feature maps that are responsive to changing object scales.

Head: The head of YOLOv8 is responsible for categorizing items inside bounding boxes and anticipating them. It generates forecasts using the Neck's fine-tuned feature maps. The head makes following predictions for every grid cell in the feature maps:

i) Probabilities of classes: An object's probability of belonging to each class (person, automobile, etc.) is represented by a vector
ii) Coordinates of the bounding box: Forecasts for the bounding box's height, width, and center (x, y), frequently parameterized in relation to the grid cell and anchor boxes
iii) Objectness score: The objectness score represents the probability of an object, rather than background, presented at the bounding box

YOLOv8 can employ anchor-free designs or anchor-based designs, like the earlier YOLO iterations. The anchor-based methods are more effectively predicts bounding boxes by using specified anchors. The anchor-free method streamlines the prediction process having the network regress bounding box coordinates directly rather than using pre-established anchors. The head converts the feature maps into outputs, such as the position, confidence, and object category. It generates unprocessed predictions for every area of the image, making it the central component of the object detection pipeline.

Post-processing: To refine the Head's raw predictions and turn them into final object detection, post-processing is essential. By suppressing all the boxes with the greatest confidence score and elimination of Non-Maximum Suppression (NMS) in terms of multiple bounding boxes for the same object. After choosing the box with the highest objectiveness score, the algorithm eliminates any boxes that overlap this box significantly as determined by Intersection over Union:

i) Confidence threshold: Because they are likely to be false positives, predictions with an objectiveness score below a predetermined threshold are eliminated. This guarantees that only detection with a high degree of confidence is kept
ii) Bounding Box Adjustment: Depending on the resolution of the finished feature map and the predictions made by nearby grid cells, the bounding boxes that the Head predicts are frequently slightly modified. This increases the precision of localization

Experimental Setup

Initially, we have trained the YOLOv8 model on our system i.e., CPU-based, but it took a very long time for execution, meaning that we were not able to completely run the model til the last epochs. Therefore, to perform all the experiments, we have used Google Colab Pro, which provides GPU support that gives a GPU Tesla T4 with a CUDA configuration of 12.40 and 13 GB of RAM. However, by default, Google Colab provides an Intel Xeon CPU with 13GB of RAM. Also, it used the latest version of Python 3.11.12, for execution of the model. On the other hand, we have tested the same images on the Tensorflow-based object detection Faster R-CNN model which is already trained on different image sizes, such as 640×640, 1024×1024, etc. The Faster R-CNN model used the *.xml format for annotation, however, the YOLOv8 model, used the *.txt file format. Therefore, in order to these two models, we have performed the experiments on both the environments and compared the results. In the next section, we will discuss the detection outcomes of both the models.

Results and Discussion

This section discusses the result of the Faster R-CNN and YOLOv8 model, respectively. Initially, we have some experiments on Faster R-CNN model with different image size 640×640, and 1024×1024 with the mAP of 0.60 and 0.61, respectively. The training performed starting from 400 to 2000 steps on the given dataset. The results are given in Figs. 3, and 4, in which model, successfully detected some diseased instances and missed some diseased portion. Figs. 5, and 6, show the examples where this model is failed to detect the diseased objects. Thereafter, we have chosen YOLOv8 model, which was trained on a dataset that had 190 labeled examples from three types of diseases: Leaf rust (16 images), yellow rust (23 images), and powdery mildew (24 images), along with actual and predicted bounding boxes.

Fig. 3: Faster R-CNN object detection results

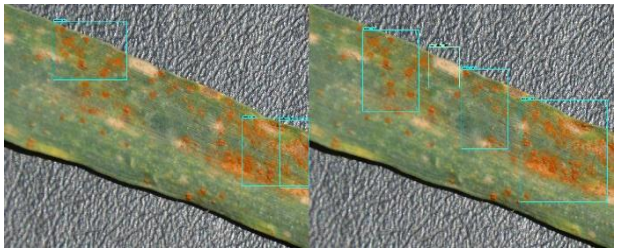


Fig. 4: Faster R-CNN detection performance evaluation

Fig. 5: Faster R-CNN Model Detection Failure Analysis

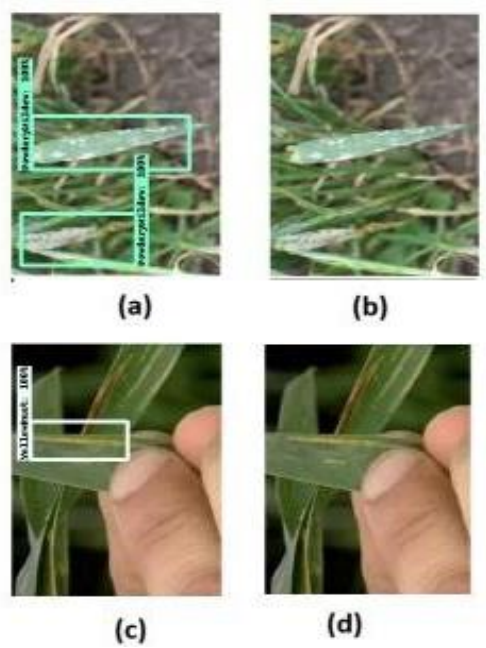


Fig. 6: Examples of Faster R-CNN Model Failures to Detect Objects (a,c) Actual Detections (b,d) Predicted Detections

Table 3 shows the mean average accuracy (mAP) values obtained from training over 400, 800, 1200, and 1600 epochs were 0.83, 0.99, 0.99, and 0.99, respectively. Randomly, we started the training on 400 and 800 epochs, there was the largest improvement in performance, with the mAP rising from 0.83 to 0.99. The mAP values improved after 800 epochs, albeit more slowly.

Table 3: Comparison of mAP results on different thresholds values at various epochs

Class	Images	Instances	Box (P)	R	mAP50	mAP50-95
400 Epochs						
All	63	190	0.88	0.71	0.83	0.59
PM	24	116	0.9	0.66	0.79	0.52
YR	23	49	0.84	0.82	0.86	0.63
LR	16	25	0.88	0.64	0.83	0.64
800 Epochs						
All	63	190	0.98	0.94	0.99	0.86
PM	24	116	0.98	0.92	0.98	0.8
YR	23	49	0.98	0.94	0.98	0.87
LR	16	25	0.98	0.96	0.91	0.9
1200 Epochs						
All	63	190	0.97	0.98	0.99	0.9
PM	24	116	0.99	0.97	0.99	0.86
YR	23	49	0.98	0.98	0.99	0.9
LR	16	25	0.97	1	0.99	0.95

From 1200 to 1600 epochs (0.994 to 0.995), which approximates to 0.99, there are slight increases. The outcomes show that YOLOv8 can efficiently learn the complex visual patterns associated with plant diseases from a comparatively small sample. The model improves quickly between 400 and 800 epochs, suggesting effective learning in the early phases of training. Since the mAP of 0.99 shows that YOLOv8 has learned most of the important features for accurate classification, the model is nearly at its best performance by 800 epochs. In contrast,

the performance improvements after 800 epochs were minor, with mAP rising about 0.009 throughout the 800–1600 epochs. This indicates that the benefits of more training become less, indicating that quitting early at 800 or 1200 epochs would be the best option for computing efficiency without significantly compromising accuracy. YOLOv8 architecture appears to handle small datasets particularly well when compared to other object detection models like Faster R-CNN. This is probably because of its efficient end-to-end processing and anchor-free design, which enable greater generalization and faster convergence. In real-world applications such as smart agriculture and plant health monitoring, this result is in line with the objectives of optimizing computer resources and maximizing accuracy for plant disease identification. Figs. 7 and 8 show the final detection results for the three wheat fungal diseases-leaf rust, powdery mildew, and yellow rust-that were detected and classified as multi-diseases in this study using the YOLOv8 model. We trained the machine using a large dataset of tagged wheat leaf images showing signs of various diseases. By enabling real-time disease detection and classification, the model was able to accurately identify the type of disease present in addition to locating infected patches on wheat leaves. To ensure that only the most reliable forecasts were considered, the model filtered out low-confidence detections using a 50% confidence criterion. This study used the YOLOv8 model to detect and classify three fungal diseases of wheat as multi-diseases: Leaf rust, powdery mildew, and yellow rust. An extensive collection of tagged wheat leaf images displaying these diseases' symptoms was used to train the machine. The model accurately identifies the type of diseases, location and diseased patches on wheat leaves, which was necessary to enable real-time detection and categorization of these diseases. The researchers used a 50% confidence criterion to ensure that only the most trustworthy predictions were considered, effectively weeding out low-confidence detections. The YOLOv8 model demonstrated the ability to correctly classify multiple diseases when they manifested on a single wheat leaf at the same time. When two diseases coexist on the same plant, as leaves with both yellow rust and leaf rust, the model was able to identify and categorize them. YOLOv8 demonstrated strong multi-disease classification abilities, delivering high-confidence predictions for each disease class when they coexisted and performing consistently across several test images. Considering all factors, the model has demonstrated significant potential for practical applications in automated crop disease management, offering precise detection and multi-disease classifications with a 50% confidence level exceeding.

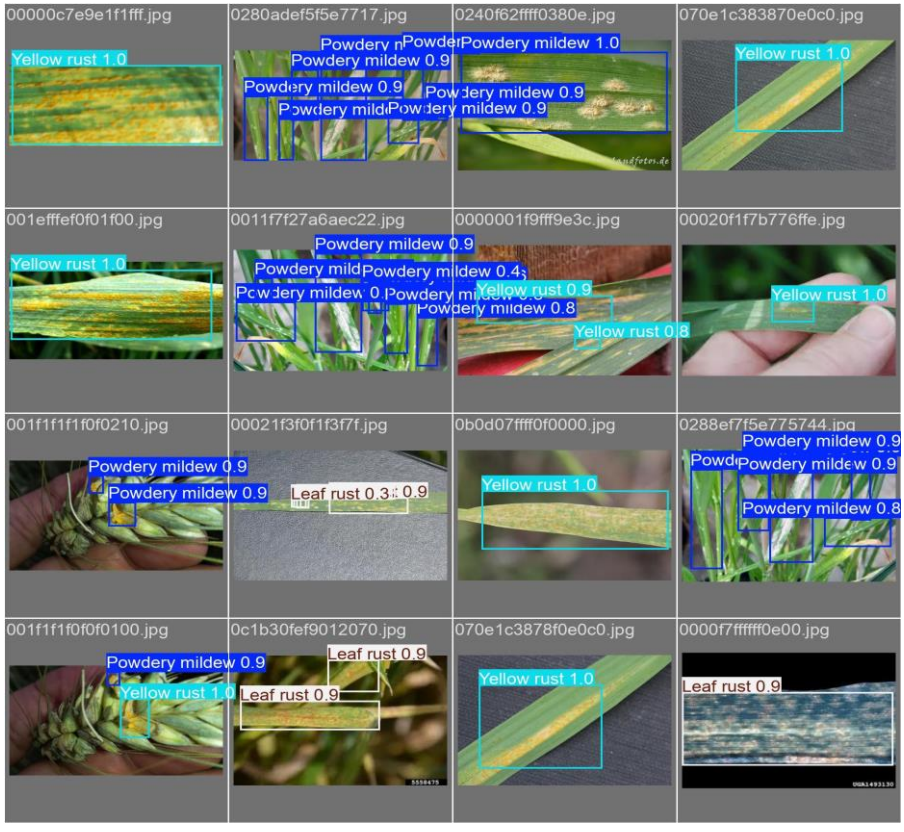


Fig. 7: Model output from YOLOv8 showing object detection

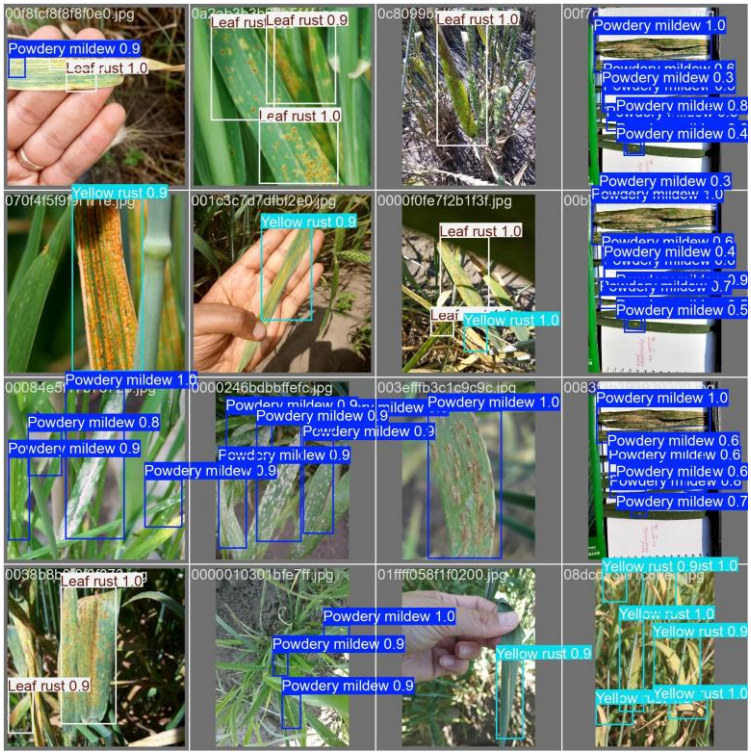


Fig. 8: Demonstrating YOLOv8's detection performance

Further, we have compared the results using different performance metrics such as confusion matrix, precision-confidence curve, recall-confidence curve, etc. The model's performance in object detection has been evaluated using metrics such as mAP, precision, and recall. Therefore, statistical significance tests are necessary to determine. The detailed description about these metrics is as given below.

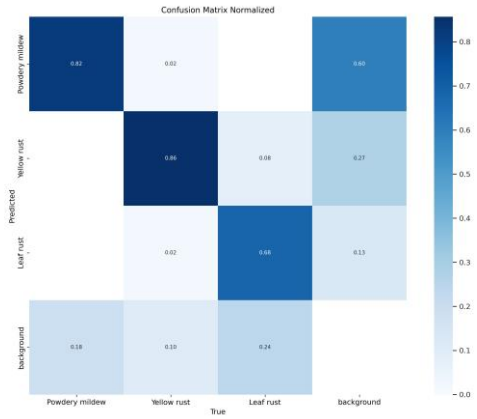
Confusion Matrix Analysis

Fig. 9 shows the confusion matrix results on different epochs, i.e., 400, 800, 1200, and 1600. As the number of training epochs increases, the confusion matrix findings show a distinct and consistent improvement in classification performance. This is especially true for leaf rust, which starts at a relatively low precision of 0.68 at 400 epochs but quickly improves to 0.96 at 800 epochs. This improvement is in line with previous findings in mean average precision (mAP), which show that the model performs significantly better between 400 and 800 epochs. The model performed moderately for 400 epochs, with precision values ranging from 0.68 for Yellow Rust to 0.86 for Leaf Rust. It also implies that the model is not yet fully tuned and is still learning to differentiate between the disease classes, particularly for leaf rust, which appears to be more challenging to identify at lower epochs. The precision values for each of the three classes greatly improve by 800 epochs, achieving remarkable consistency between classes. Leaf and yellow rust attain 0.96, while powdery mildew obtains 0.97. This result shows that the model has significantly improved its detection performance across every disease class and is now able to correctly identify disease features. The model achieves near-perfect precision for epochs = 1200 and 1600, with powdery mildew coming in slightly behind 0.99 and yellow & leaf rust at 1.00. The model has largely converged by 1200 epochs, as evidenced by the small performance difference between 1200 and 1600 epochs; further training provides no improvement. From a performance perspective, it might not be necessary to train the model for more than 1200 epochs, since it already achieves almost perfect classification. Plant disease detection works very well with the YOLOv8 model, as shown in this analysis, showing quick improvements in accuracy during the first 800 epochs and stabilizing by 1200 epochs. The findings indicate that an early stop at 1200 epochs would probably provide the best trade-off between computing efficiency and model accuracy. Also, the strength of the YOLOv8 design is shown by how well the model can work with different disease types, even when using a smaller amount of data.

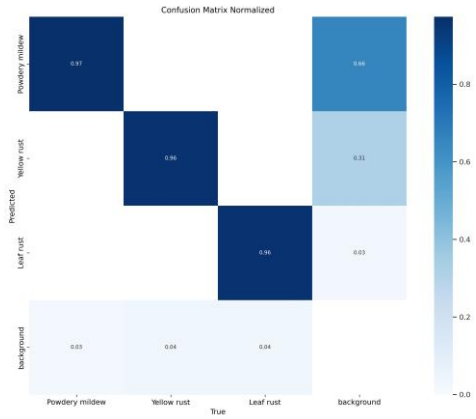
Precision-Confidence Curve Analysis

The relationship between precision of the model and its confidence threshold is reflected in the precision-confidence curve, which offers insights into the model's

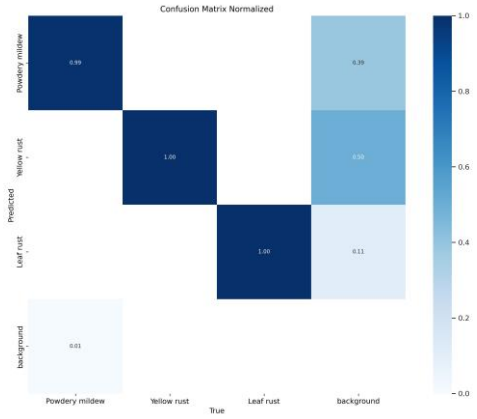
performance at various levels of certainty. Fig. 10 presents the precision-confidence curve analysis, which reveals a clear performance drop at 1600 epochs; additionally, the accuracy values across different epochs demonstrate an overall improvement in the model's ability to accurately identify and categorize the three diseases over time.



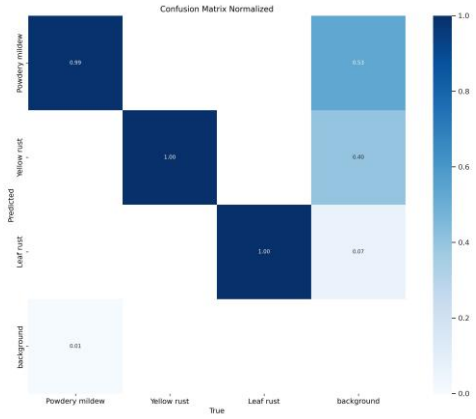
(a) Epochs = 400



(b) Epochs = 800

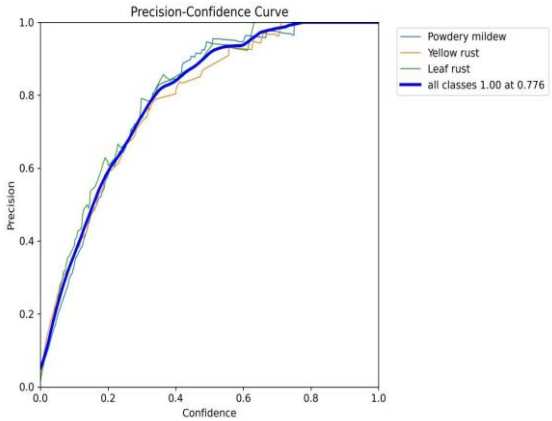


(c) Epochs = 1200

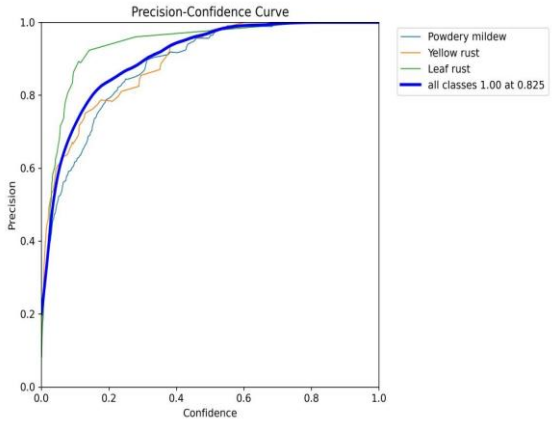


(d) Epochs = 1600

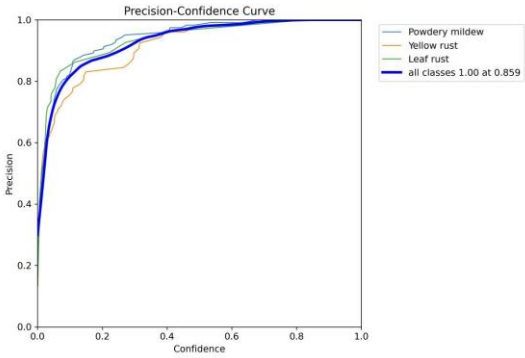
Fig. 9: Epoch-spanning confusion matrices that show variations in classification precision and error trends



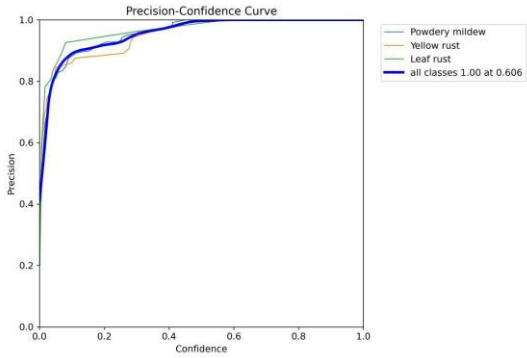
(a) Epochs = 400



(b) Epochs = 800



(c) Epochs = 1200



(d) Epochs = 1600

Fig. 10: Precision evaluation over epochs shows model consistency and development

The model's precision at 200 epochs was 0.78, meaning that even while it could accurately forecast events in about 78% of cases, there was still a good deal of space for improvement. At this point, the model's reasonable but not satisfactory classification performance is probably the result of insufficient exposure to the training set. Since the model is still having difficulty generating consistently highly accurate predictions, the precision-confidence curve at this point is probably flatter, with lower precision at higher confidence thresholds. Precision achieved 0.83 after 400 epochs, indicating a significant improvement as the model learned more about the characteristics of condition. This indicates that further training produced fewer false positives and more accurate detection. While the model may still show some degree of inconsistency in classifying the more complex or visually comparable diseases, the precision-confidence curve at this point would be more stable, with better precision across most confidence levels.

The model achieved greatest degree of precision of classification in this evaluation by 800 epochs, with a precision of 0.86. This shows that powdery mildew, yellow rust, and leaf rust could all be recognized by the model. At this stage, the model had grown increasingly confident in its predictions while maintaining accuracy, and the precision-confidence curve would probably show a strong and consistent precision throughout a wide range of confidence levels. Additionally, the earlier mAP results, which show notable improvements between 400 and 800 epochs, with consistent precision improvement. Significantly, precision decreased from earlier findings to 0.61 at 1600 epochs. The drop in precision could be due to overfitting, which happens when the model becomes too closely adjusted on the training data and struggles to perform well on new data. When the model is overfitted, it may lose confidence in its predictions, especially when the test data contains small variances. The model's overconfidence in inaccurate predictions would cause the precision-confidence curve to fluctuate more at this point, with low precision value at higher confidence levels.

Recall-Confidence Curve Analysis

Fig. 11 shows the results of the recall-confidence curve for detecting the wheat fungal diseases. The recall-confidence curve shows the range of confidence levels at which the model finds real positives. A high recall means that the powdery mildew, yellow rust, and leaf rust classes' majority of important instances can be identified by the model with a high degree of accuracy. We analyze the recall performance at various epochs. The model's recall value at 200 epochs was 0.95, meaning that 95% of true positive cases across every disease classification were properly identified. Although this is a decent first performance, the model probably missed some real positives, particularly for classes that might have comparable or complicated visuals. At this point, the recall-confidence curve should appear to perform well at lower confidence thresholds but should perform somewhat worse at higher thresholds value, when the model becomes more cautious and might miss some true cases. The model obtained a perfect recall of 1.00 after 400 epochs of training, which indicates that it successfully identified every true positive in every class. This represents a noteworthy advancement, demonstrating that the model has acquired the ability to identify every occurrence of sickness inside the dataset. At this stage, the model has grown more confident in its ability to recognize true positives without missing important instances; therefore, the recall-confidence curve would probably show a very high recall even at higher confidence thresholds. The recall was stable at 1.00 at 800 epochs, meaning that the model was still able to accurately identify every incident of leaf rust, yellow rust, and powdery mildew. This consistently high recall number implies that the model had learned as much as it could and was no longer missing any really positive cases. With high recall values at all confidence levels, the recall-confidence curve would stay relatively flat, demonstrating the model's excellent performance across a range of thresholds.

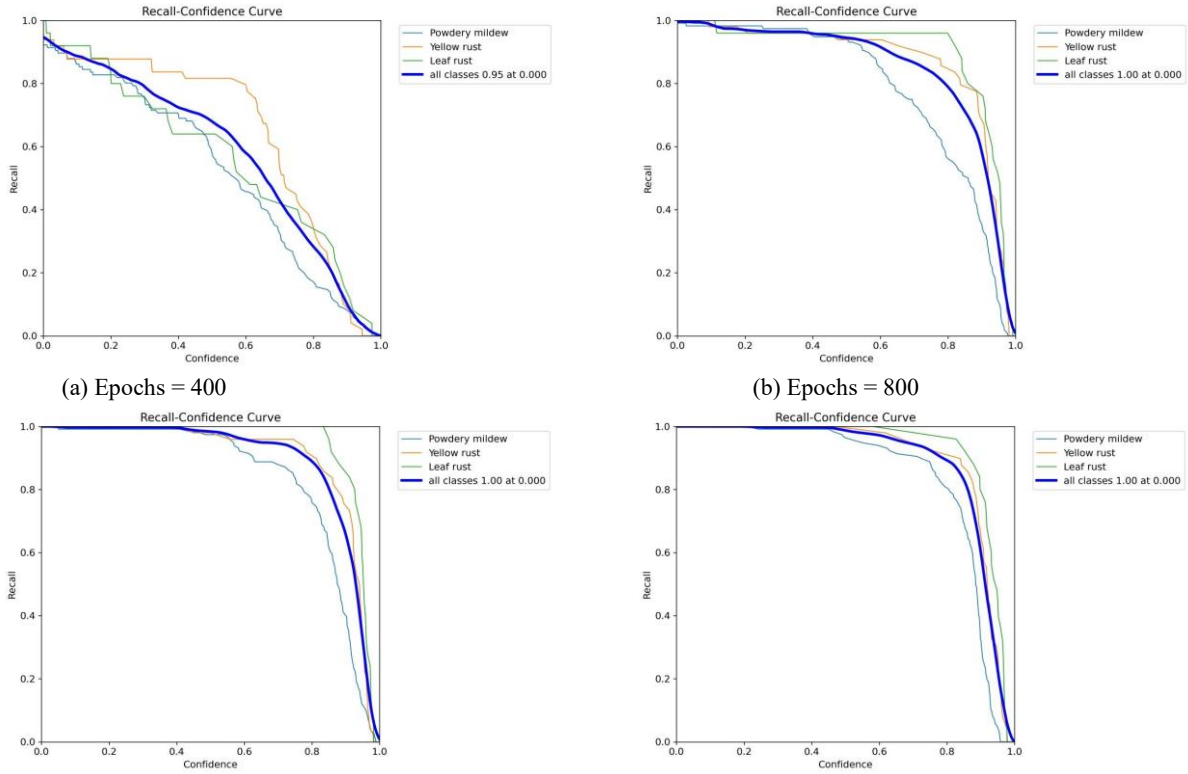


Fig. 11: Analysis of recall curves epoch-wise shows model performance trends

The recall remains at 1.00 even at 1600 epochs, indicating that the model was still able to identify all true positives. Even though recall is still perfect, it's crucial to keep in mind that this does not always mean that performance as a whole is at its best. The precision study conducted before revealed that overfitting caused a considerable decline in precision at 1600 epochs. At all confidence levels, the recall-confidence curve would still show excellent recall; however, the model may be recognizing erroneous positives in addition to real positives, which would reduce precision.

Precision-Recall Curve Analysis

Fig. 12 represents the comparison of precision and recall curves at different epochs. When analyzing imbalanced datasets or concentrating on reducing false positives and increasing true positives, the precision-recall (PR) curve is a helpful indicator for assessing model performance. Trade-offs between precision and recall show how successfully a model classifies true positives (recall) while retaining the proper ratio of false positives (precision). The PR curve analysis for every moment is shown below.

The model's precision at 200 epochs was 0.83. Although the classification performance is reasonably excellent, this result indicates that the model has difficulty in differentiating across classes, which results in a moderate amount of false positives. At this stage, we expect the accuracy-recall curve to vary a bit, with some lower accuracy when the recall is high, indicating that the model can find many cases but makes some mistakes in classification. Due to a relatively high false positive rate at this early training stage, the curve might suggest a mild rise before possibly flattening out at higher recall values. The model's recall achieved 1.00, and its precision improved to 0.99 by 400 epochs. This represents a major improvement in performance as the model continues to retain excellent recall and high precision. At this point, the PR curve would be far smoother and maintain a greater level of precision even at high recall levels. It indicates that the model has improved to the point where it can accurately identify the three disease classifications with the fewest false positives. The curve would exhibit outstanding precision throughout a wide range of recall values, rising abruptly and remaining at a high level. The recall remained at 1.00 while the precision developed to 0.994 after 800 epochs. This shows that, with extremely few false positives and almost perfect categorization in every case, the model has almost attained its peak performance in terms of balancing precision and recall. At this time, there would be little departure from ideal performance, as both the accuracy and recall values would be getting close to their maximums on the accuracy-recall curve.

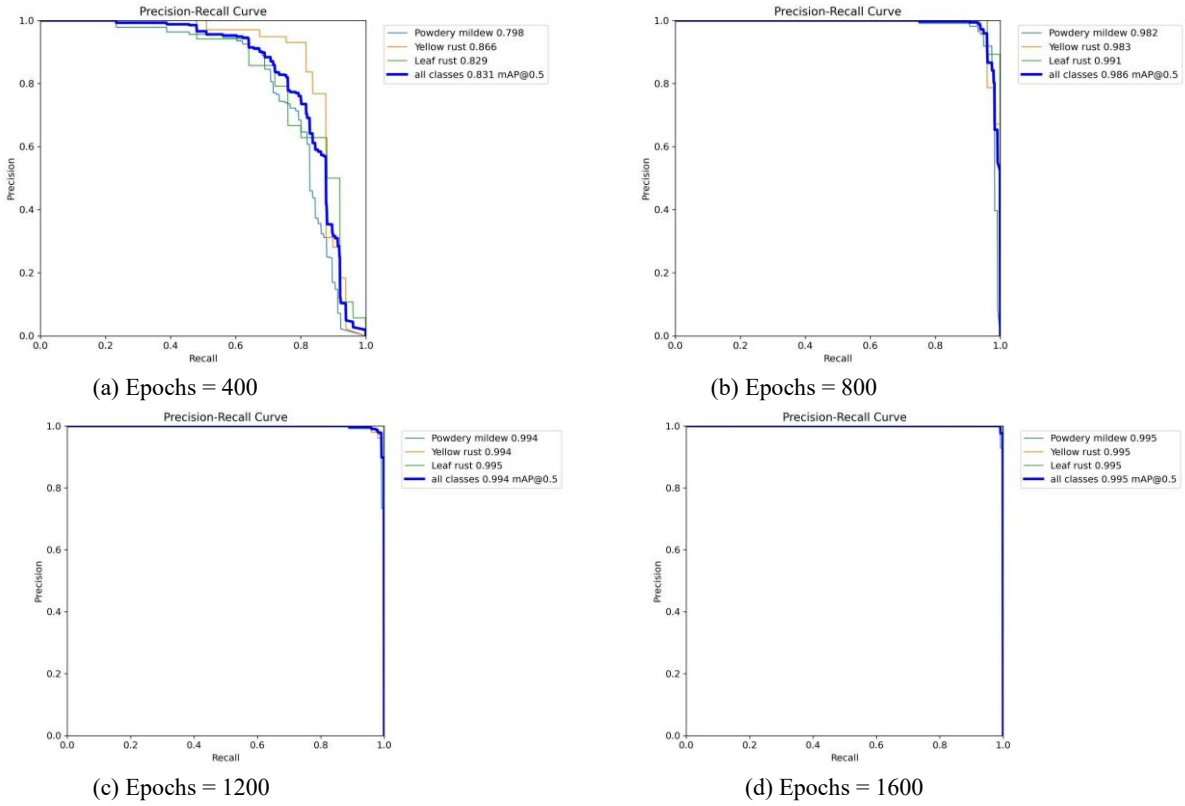


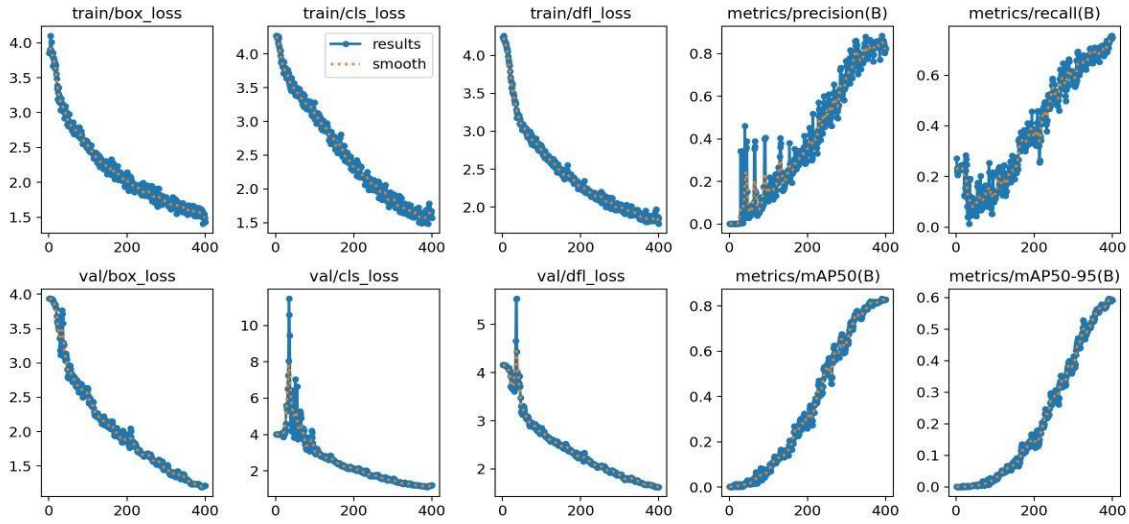
Fig. 12: Precision-Recall curve comparisons across different model configurations

The model's performance would be nearly perfect as the curve approaches the ideal step function, where precision stays high as recall approaches 1.00.

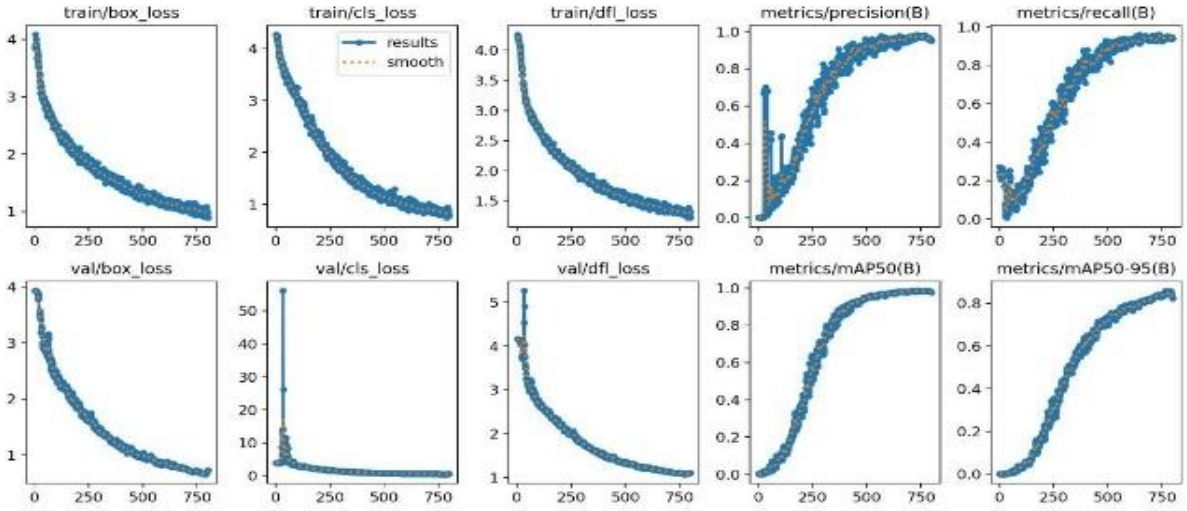
Precision reached 0.99 at 1600 epochs, a marginal improvement over 800 epochs. With the increase of epochs resulting in diminishing rewards, but at this point the marginal precision improvement shows that the model has successfully converged its learning. The model continues to detect all true positives without missing any, any true positives as an evidenced by the recall of 1.00, having still flawless. However, the relatively small improvement noticed earlier may be explained by having slight overfitting affect. The PR-curve would still show almost perfect performance, in comparison to 800 epochs, the slight increase in precision does not substantially alter the curve's general shape. At all recall levels, the curve stays flat with precision near 1.00, demonstrating the model's reliable ability to categorize almost all cases accurately. The model experiences challenges balancing recall and precision at 200 epochs, which increases the number of false positives. It attains high precision and perfect recall by 400 epochs and keeps this equilibrium until 800 epochs, at which point precision is almost perfect. Additional training after 800 epochs results in overfitting and does not enhance performance. Consequently, the 800-epoch model provides optimal performance without overfitting by providing the best trade-off between precision and recall.

Loss Comparison

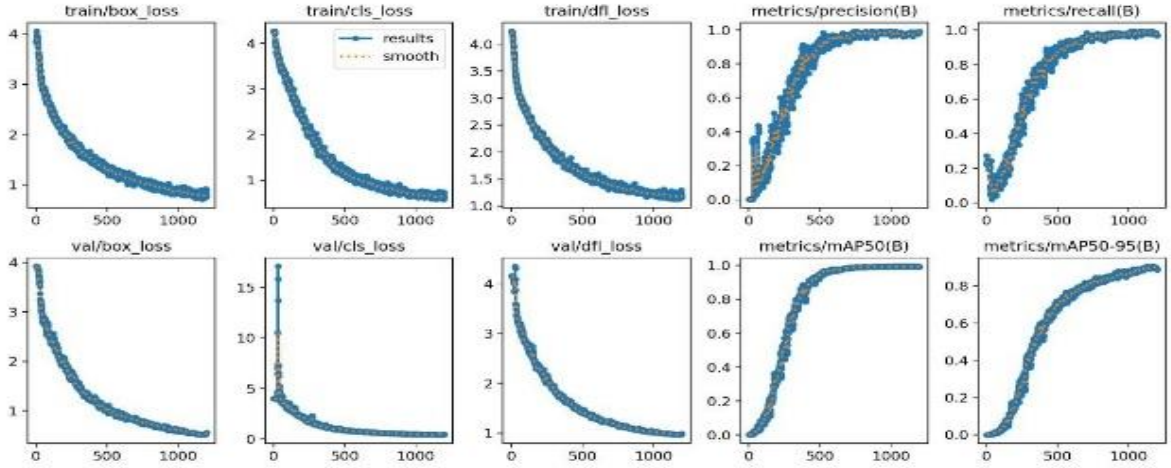
Fig. 13 shows how the loss of the YOLOv8 model changed as it was trained for 400, 800, 1200, and 1600 epochs in this study, with the loss consistently decreasing as the number of epochs increased. In the first 400 epochs, the model quickly learned from the training data, which improved how it identified features, predicted bounding boxes, and classified classes, leading to a significant drop in overall loss. The loss kept declining by the 800th epoch, but the rate of reduction slowed, suggesting that the model was going close to convergence. In the first 400 epochs, the model quickly learned from the training data, which improved how it identified features, predicted bounding boxes, and classified classes, leading to a significant drop in overall loss. The loss kept declining by the 800th epoch, but the rate of reduction slowed, suggesting that the model was getting close to convergence. The consistent drop in loss at this stage showed that the YOLOv8 architecture successfully learned from the data, modifying its biases and weights to enhance performance. The loss decreased, although much more slowly, as the number of epochs increased to 1200 and 1600. The loss had almost stagnated by the 1200th epoch, indicating that the model's ability to reduce missed detection and had been fully realized.



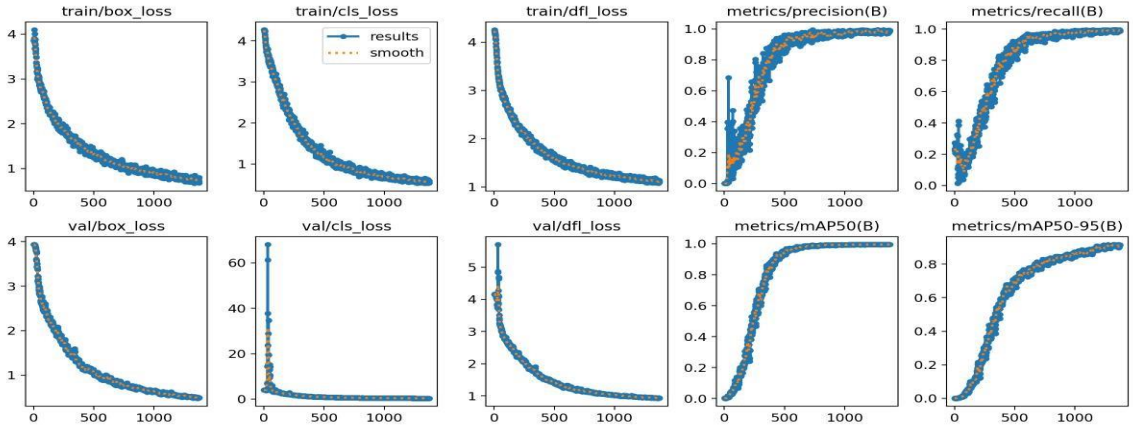
(a) Epochs = 400



(b) Epochs = 800



(c) Epochs = 1200



(d) Epochs = 1600

Fig. 13: Comparison of training epoch losses

Table 4: Comparison of YOLOv8 model with the state-of-the-art techniques

References	Total no. of images	Methodology	mAP
Venkataram etal. (2019)	anan36,148	YOLOv3	0.96
Roy and Bhaduri, (2021)	2,000	YOLOv4	0.91
Proposed Model	63	YOLOv8	0.99

Only with the slight improvements were seen after 1200 epochs, and the overall loss stabilized close to its lowest value. Significantly, the training and validation losses showed comparable trends, suggesting strong generalization and lowering the possibility of overfitting. Since improvements were minimal after 1200 epochs, the model's ability to learn and generalize was confirmed by the consistent drop in loss throughout the long training period.

Consequently, in Table 4 we have compared the result of YOLOv8 model with other YOLO models implemented by the other researchers. In which, it is stated that YOLOv8 model gives the best result in smaller amount of the dataset when compared with the state-of-the-art. However, the model's resilience in real-world situations is limited by the absence of biological and environmental variability, such as changes in leaf orientation, illumination, and disease severity.

Reliability and wider applicability were limited by the analysis of only three diseases and the lack of cross validation and external testing. Larger and more varied datasets should be used in future research.

Limitations

In this study, only 63 images were used and the pictures were classified into three classes, namely, leaf rust, yellow rust and powdery mildew. To our knowledge, data augmentation strategies (that is, rotations, flips, cropping, color jittering, and GAN-based synthesis) were not used. The methods mentioned are characteristics of computer vision and can be used to augment the diversity of the dataset and enhance model generalization when facing a scenario where only a small amount of data is present. Even though we have provided statistical results carried out on the performance measures (mAP, precision, and recall) in section titled Results and Discussions, we did not provide a formal statistical test of significance. Such tests are the paired t-tests, Wilcoxon signed-rank test, bootstrap confidence intervals or effect size computations. These tests are used to identify whether the differences in those performances are indeed significant or the effects are just artifacts of noise. We know that a possible shortcoming of our experiment is the fact that not all augmentations were used which might have restricted the robustness of the model, and we have mentioned the limited size of the dataset used which is a limitation that we have identified. And finally, the present publication still lacks the definition of the following measures that will be taken, but we plan to expand the set of pictures and make it more varied in terms of the features and classes in the future work that we are going to conduct to achieve the more solid generalizations estimates and reduce the threat of overfitting.

Conclusion and Future Work

The purpose of this study is to determine how well the YOLOv8 model can identify and categorize plant diseases, namely Powdery Mildew, Yellow Rust, and Leaf Rust, using a dataset of 10 images. When the model's performance was assessed at several epochs (i.e., 400, 800, 1200, and 1600), the mean average precision (mAP) increased from 0.83 to 0.99. The

model got nearly perfect scores for all classes (0.97, 0.96, and 0.96) at 800 epochs, as seen by the confusion matrix findings, which showed significant gains in classification accuracy. The model continued to perform at this high level at 1200 and 1600 epochs, indicating remarkable consistency in its detection abilities. Overall, the results show that the YOLOv8 model performs best at 800 epochs and is a highly useful tool for detecting plant diseases. The findings emphasize how crucial it is to balance training duration in order to prevent overfitting and preserve good recall and precision. In order to further strengthen identification capabilities in various agricultural contexts, future research could investigate improved data augmentation methodologies and comparisons with other object detection algorithms. In order to enhance performance, future research should concentrate on optimizing the YOLOv8 model architecture using sophisticated methods like hyper-parameter tuning and investigating various variations. The robustness and generalization of the model to new cases can be improved by implementing a variety of data augmentation techniques. The capabilities of the model would be further tested by extending the study to multi-class disease detection, in which multiple diseases emerge on the same leaf. Further information regarding YOLOv8's efficacy may also be obtained by contrasting it with alternative object detection models, such as Faster R-CNN or RetinaNet. Lastly, the model would be more useful for real-world applications if it were optimized for real-time deployment in environments like agriculture.

Acknowledgment

For the current work we thank to the founder of YOLOV8 model for proposing this model that really makes the detection process more efficient.

Funding Information

There is no funding available for the current work.

Author's Contributions

Shivani Sood: Conceptualization, Methodology, Investigation, Write - original draft preparation, methodology.

Shallu Duggal: Conceptualization, Write - original draft preparation. Monika Sethi: Investigation, Methodology, Writing - review and edited.

Chander Prabha: Review and editied, analysis of results.

Prakash Srivastava: Conceptualization, review and analysis.

Mohammad Zubair Khan: Resources, Software, Editing, Review and Analysis.

Amna Bamaqa: Conceptualization, Software, validation.

Dataset Availability

Dataset used in this study can be download from "http://wfd.sysbio.ru/".

Abdulaziz Alblwi: Investigation, resources, Analysis of results.

Ethics

This is an original article. The corresponding author confirms that all authors have reviewed and approved the manuscript. There are no ethical issues involved.

References

Agbulos, Ma. K., Sarmiento, Y., & Villaverde, J. (2021). Identification of Leaf Blast and Brown Spot Diseases on Rice Leaf with YOLO Algorithm. *2021 IEEE 7th International Conference on Control Science and Systems Engineering (ICCSSE)*, 307–312. https://doi.org/10.1109/iccsse52761.2021.9545153

Alahmari, F., Naim, A., & Alqahtani, H. (2023). E-Learning Modeling Technique and Convolution Neural Networks in Online Education. *Taylor and Francis Group*, *1*, 261–295. https://doi.org/10.1201/9781003393030-10

Ampatzidis, Y., De Bellis, L., & Luvisi, A. (2017). iPathology: Robotic Applications and Management of Plants and Plant Diseases. *Sustainability*, *9*(6), 1010. https://doi.org/10.3390/su9061010

Anjanadevi, B., Charmila, I., Ns, A., & Anusha, R. (2020). An improved deep learning model for plant disease detection. *International Journal of Recent Technology and Engineering*, *8*(6), 5389–5392.

Bond, J., & Liefert, O. (2016). *Wheat outlook*.

Duro, D. C., Franklin, S. E., & Dubé, M. G. (2012). A comparison of pixel-based and object-based image analysis with selected machine learning algorithms for the classification of agricultural landscapes using SPOT-5 HRG imagery. *Remote Sensing of Environment*, *118*, 259–272. https://doi.org/10.1016/j.rse.2011.11.020

Eunice, J., Andrew, J. S., Popescu, D. E., Chowdary, M. K., & Hemanth, J. (2022). Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural Applications. *Agronomy*, *12*(10), 2395. https://doi.org/10.3390/agronomy12102395

Kartikeyan, P., & Shrivastava, G. (2021). Review on Emerging Trends in Detection of Plant Diseases using Image Processing with Machine Learning. *International Journal of Computer Applications*, *174*(11), 39–48. https://doi.org/10.5120/ijca2021920990

Kaur, P., Harnal, S., Gautam, V., Singh, M. P., & Singh, S. P. (2022). An approach for characterization of infected area in tomato leaf disease based on deep learning and object detection technique. *Engineering Applications of Artificial Intelligence*, *115*, 105210. https://doi.org/10.1016/j.engappai.2022.105210

Khan, H., Haq, I. U., Munsif, M., Mustaqeem, Khan, S. U., & Lee, M. Y. (2022). Automated Wheat Diseases Classification Framework Using Advanced Machine Learning Technique. *Agriculture*, *12*(8), 1226. https://doi.org/10.3390/agriculture12081226

Kurmi, Y., Gangwar, S., Chaurasia, V., & Goel, A. (2022). Leaf images classification for the crops diseases detection. *Multimedia Tools and Applications*, *81*(6), 8155–8178. https://doi.org/10.1007/s11042-022-11910-7

Lee, D. I., Lee, J. H., Jang, S. H., Oh, S. J., & Doo, I. C. (2023). Crop Disease Diagnosis with Deep Learning-Based Image Captioning and Object Detection. *Applied Sciences*, *13*(5), 3148. https://doi.org/10.3390/app13053148

Li, Z., Fang, X., Zhen, T., & Zhu, Y. (2023). Detection of Wheat Yellow Rust Disease Severity Based on Improved GhostNetV2. *Applied Sciences*, *13*(17), 9987. https://doi.org/10.3390/app13179987

Lorrain, C., Gonçalves dos Santos, K. C., Germain, H., Hecker, A., & Duplessis, S. (2019). Advances in understanding obligate biotrophy in rust fungi. *New Phytologist*, *222*(3), 1190–1206. https://doi.org/10.1111/nph.15641

Luo, Z., McTaggart, A., & Schwessinger, B. (2024). Genome biology and evolution of mating-type loci in four cereal rust fungi. *PLOS Genetics*, *20*(3), e1011207. https://doi.org/10.1371/journal.pgen.1011207

Morbekar, A., Parihar, A., & Jadhav, R. (2020). Crop Disease Detection Using YOLO. *2020 International Conference for Emerging Technology (INCET)*, 1–5. https://doi.org/10.1109/incet49848.2020.9153986

Orchi, H., Sadik, M., Khaldoun, M., & Sabir, E. (2023). Real-Time Detection of Crop Leaf Diseases Using Enhanced YOLOv8 algorithm. *2023 International Wireless Communications and Mobile Computing (IWCMC)*, 1690–1696. https://doi.org/10.1109/iwcmc58020.2023.10182573

Peng, J. H., Sun, D., & Nevo, E. (2011). Domestication evolution, genetics and genomics in wheat. *Molecular Breeding*, *28*(3), 281–301. https://doi.org/10.1007/s11032-011-9608-4

Roy, A. M., & Bhaduri, J. (2021). A Deep Learning Enabled Multi-Class Plant Disease Detection Model Based on Computer Vision. *AI*, *2*(3), 413–428. https://doi.org/10.3390/ai2030026

Roy, A. M., Bose, R., & Bhaduri, J. (2022). A fast accurate fine-grain object detection model based on YOLOv4 deep neural network. *Neural Computing and Applications*, *34*(5), 3895–3921. https://doi.org/10.1007/s00521-021-06651-x

Saleem, M. H., Potgieter, J., & Arif, K. M. (2019). Plant Disease Detection and Classification by Deep Learning. *Plants*, *8*(11), 468. https://doi.org/10.3390/plants8110468

Sereda, I., Danilov, R., Kremneva, O., Zimin, M., & Podushin, Y. (2023). Development of Methods for Remote Monitoring of Leaf Diseases in Wheat Agrocenoses. *Plants*, *12*(18), 3223. https://doi.org/10.3390/plants12183223

Slimani, H., Mhamdi, J. E., & Jilbab, A. (2023). Artificial Intelligence-based Detection of Fava Bean Rust Disease in Agricultural Settings: An Innovative Approach. *International Journal of Advanced Computer Science and Applications*, *14*(6), 119–121. https://doi.org/10.14569/ijacsa.2023.0140614

Sood, S., & Singh, H. (2020). An implementation and analysis of deep learning models for the detection of wheat rust disease. *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, 341–347. https://doi.org/10.1109/iciss49785.2020.9316123

Sood, S., & Singh, H. (2021). Computer Vision and Machine Learning based approaches for Food Security: A Review. *Multimedia Tools and Applications*, *80*(18), 27973–27999. https://doi.org/10.1007/s11042-021-11036-2

Sood, S., & Singh, H. (2022). Effect of Kernel Size in Deep Learning-Based Convolutional Neural Networks for Image Classification. *ECS Transactions*, *107*(1), 8877–8884. https://doi.org/10.1149/10701.8877ecst

Sood, S., & Singh, H. (2024). A comparative study of grape crop disease classification using various transfer learning techniques. *Multimedia Tools and Applications*, *83*(2), 4359–4382. https://doi.org/10.1007/s11042-023-14808-0

Sood, S., Singh, H., & Jindal, S. (2022). Rust Disease Classification Using Deep Learning Based Algorithm: The Case of Wheat. *Food Systems Resilience*, 1–79. https://doi.org/10.5772/intechopen.104426

Sood, S., Singh, H., & Malarvel, M. (2021). Image quality enhancement for Wheat rust diseased images using Histogram equalization technique. *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, 1035–1042. https://doi.org/10.1109/iccmc51019.2021.9418023

Si, Y., Liu, G., & Feng, J. (2015). Location of apples in trees using stereoscopic vision. *Computers and Electronics in Agriculture*, *112*, 68-74. https://doi.org/10.1016/j.compag.2015.01.010

Uoc, N. Q., Duong, N. T., Son, L. A., & Thanh, B. D. (2022). A novel automatic detecting system for cucumber disease based on the convolution neural network algorithm. *GMSARN Int. J*, *16*(3), 295–301.

Venkataramanan, A., Honakeri, D. K. P., & Agarwal, P. (2019). Plant disease detection and classification using deep neural networks. *Int. J. Comput. Sci. Eng*, *11*(9), 40–46.

Wang, Q., Liu, C., Xia, X., Guo, Y., & Men, H. (2024). Classification and identification of crop disease based on depthwise separable group convolution and feature fusion. *Journal of Plant Diseases and Protection*, *131*(2), 601–615. https://doi.org/10.1007/s41348-023-00826-5

Wijayanto, A. K., Junaedi, A., Sujaswara, A. A., Khamid, M. B. R., Prasetyo, L. B., Hongo, C., & Kuze, H. (2023). Machine Learning for Precise Rice Variety Classification in Tropical Environments Using UAV-Based Multispectral Sensing. *AgriEngineering*, *5*(4), 2000–2019. https://doi.org/10.3390/agriengineering5040123

Zhang, Y., Song, C., & Zhang, D. (2020). Deep Learning-Based Object Detection Improvement for Tomato Disease. *IEEE Access*, *8*, 56607–56614. https://doi.org/10.1109/access.2020.2982456