Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Optimizing CNN-YOLOv7 Models for Pepper Leaf Disease Detection and Identification

Salma Asiya Begum Shaik[1], Hussain and Syed[2]

[1]Sr. Assistant Professor, Dept of CSM, LakiReddy Bali Reddy College of Engineering, India, salma.ahu172@gmail.com
[2]Associate Professor, School of Computer Science & Engineering, VIT-AP University, India, hussain.syed@vitap.ac.in

Agriculture is an important source of economic growth in many emerging nations, including Ethiopia. Pepper cultivation is essential for food security worldwide. Peppers are subject to blight leaf disease, grey leaf spot, common rust, fruit rot, and powdery mildew. Researchers have discovered over 34 pepper pathogens, leading to a 33% reduction in crop production. Farmers have used visual inspection to detect infections, but it is inaccurate and time-consuming. Previous research has explored using image processing and DL to classify pepper plant viruses. This study presents a CNN-YOLOv7 concatenated neural network. This goal is to create a completely linked-layer classification model for pepper leaf diseases. The CNN-YOLOv7 model follows a multi-stage procedure that includes dataset collection, image processing, noise removal, segmentation, feature extraction, and classification. This process optimizes the hyperparameters of the model to achieve a balance between accuracy and computational efficiency. Additionally, it leverages transfer learning to identify subtle symptoms of disease. Various datasets have verified the model's ability to identify pepper leaf illnesses, suggesting its potential for agricultural use. This tunes the CNN-YOLOv7 model for precision and computational efficiency, allowing it to be used in real-time scenarios through various fields. This study advances computer vision for agriculture by providing a reliable mechanism for proactive plant health monitoring. The CNN-YOLOv7 hybrid model accurately diagnoses pepper leaf diseases with 98.92% and 99.02% average precision (AP) accuracy. This allows for real-time disease evaluation in field settings, which improves food security by promoting sustainable farming approaches.

Keywords: CNNs, YOLOv7, Image preprocessing, Segmentation, Feature extraction, and Classification.

1. Introduction

Developing countries' economies heavily rely on agriculture and its products. Diseases that have a negative impact on a plant's productivity, growth, and economic prosperity are prevalent. Due to their critical role in enhancing and predicting agricultural output, digital image processing and image analysis technologies have recently gained significance for plant diseases. Globally, people cultivate pepper, primarily using it as a spice. The primary objective of the present study was to categorize several varieties of peppers utilized for both commercial and culinary applications in Ethiopia. Pepper comprises 67.98% of the total cultivated vegetable acreage in Ethiopia. The agricultural sector dominates the Ethiopian economy, according to the financial accounts from 2012. Pepper is a major crop that grows well in many parts of Ethiopia and might help the country get closer to its agricultural goal of being self-sufficient in food production. Reduced crop growth, financial losses, and diminished pepper diseases all have a direct effect on both the quantity and quality of agricultural output. Figure 1 illustrates some of the factors contributing to the pepper plant's leaves falling off. The visual aspect of agricultural products is the primary quality characteristic. The product's visual appeal plays a crucial role in determining its value at the point of sale and influencing consumer buying decisions. Ensuring the development of robust and health-promoting plants is crucial in the agriculture sector, requiring thorough quality evaluation and inspection procedures (Sannakki et al. 2013).The agriculture sector suffers substantial economic and production losses because of plant diseases. Disease management presents a formidable undertaking (Rubia J and R 2021).Plants commonly observe plant disease symptoms on their stems or leaves, such as discoloration or streaks. Many plant dis-eases affect the leaves, and the most common culprits are fungi, bacteria, and viruses. These bacteria cause infections that manifest as a range of observable symptoms on the plant's stem or leaves (Singh and Misra 2017). Usually, this disease identifies these symptoms manually. Automatic identification of many diseases is now possible with the use of image processing techniques (Francis, Anto Sahaya Dhas D, and Anoop B K 2016a). Image processing is of the utmost importance when it comes to plant disease detection, as it yields optimal outcomes while minimizing human labor. The agriculture sector has the potential to utilize image processing in multiple applications. The process involves the identification of diseased leaves, stems, or fruits, the measurement of the affected region's area, and the determination of its color. Pepper agricultural is among India's most lucrative agricultural pursuits. Worldwide, black pepper is the most widely utilized condiment. Studies have shown that pepper thrives in regions with temperatures ranging from 15 to 400 degrees Celsius. Cultivating pepper plants to their full potential yields superior results (Camargo and Smith 2009; Ramanjot et al. 2023; Sankaran et al. 2010). Plant disease is one of the primary contributors to product quantity and quality deterioration. Specialist observation with the unaided eye is the typical method for plant identification and detection (Francis et al. 2016; Revathi and Hemalatha 2012).This method requires considerable time on expansive plantations or land areas. To improve product quality, this study investigates the critical role of image processing methods in the early diagnosis of plant illnesses.

This study's goal is to find out about diseases in the pepper plant using a sophisticated DL method known as YOLOv7. Figure 1 depicts a pepper plantation, emphasizing the significant difficulty of identifying plant diseases in an agricultural setting. The manual identification of plants is challenging due to the substantial time required to inspect every plant across the entire farm. Furthermore, the precision of such predictions is moderate, and the procedure for identifying damaged leaves is laborious. Hence, it is crucial to construct a model that can precisely and consistently detect illnesses with a greater degree of support and dependence. Our study introduces a method that can aid farmers in detecting infections without incurring any extra costs. Bacterial spot is the main affliction that affects the pepper plant. This disease predominantly affects the pepper leaf. Figure 2a depicts a leaf in a healthy state, while Figure 2b depicts a leaf in an unhealthy state.

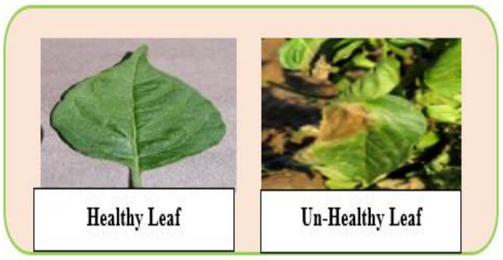


Figure 1 Reasons for Falling off Pepper Plants Leaves. Figure 2a Healthy Leaf Figure 2b Unhealthy Leaf

2. Literature Survey

Pepper leaf infections have been recently identified by researchers using deep learning algorithms. Wu and colleagues (2020) investigated an integrated neural network that utilizes CNNs to identify and classify the severity of bacterial spot diseases in pepper plants. This network had a 95.34% accuracy rate, making it the most accurate. In 2020, Yin et al. and Gu et al. demonstrated how to use transfer learning-based deep features to identify illnesses and pests in hot pepper photos. In 2022, the YOLOv5 algorithm discovered the bell pepper bacterial spot disease. There were clues in the farm's foliage. In their study, Mahesh and Mathew (2023) employed YOLOv3, a cutting-edge object detection algorithm, to accurately identify the development of bacterial spot disease in an image of a bell pepper plant. According to Mustafa et al. (2023), a five-layer CNN model can automatically identify diseases in pepper bell plants by analyzing leaf photographs with an accuracy of 90%. In identifying between bacterial and healthy plant leaves, this model achieves an astounding 99.99% accuracy. Large-scale pepper farming is still difficult to do, even with encouraging advancement. The dataset, which originated from a benchmark dataset, underwent extensive laboratory research. To identify pepper leaf infections, only a small quantity of real-time data is required. Comparatively speaking to other crop diseases, pepper infections are more difficult to identify because of their similar symptoms within and between categories (Wu et al., 2020).

Advanced deep learning algorithms, including the SSD model (Sinan, 2020), the YOLO model (Ponnusamy et al., 2020; Ganesan and Chinnappan, 2022), and others, are currently categorizing and detecting leaf diseases. Cheng et al. (2022) suggested a more efficient version of YOLOv4, a convolutional neural network, for detecting plants and maize seedlings. They trained the model using 1800 photographs. Initially, they reduce the parameters and employ the MobileNetv3 architecture is utilized as a more efficient feature extraction backbone network, replacing the CSPDarkNet53 architecture in YOLOv4, in order to reduce resource consumption. They also use this to transfer learning and accelerate the training process. The model achieves a detection speed of 69.76 frames per second (FPS) and has a total of 8.17 million parameters. It also achieves an average precision of 89.98% (mAP). However, augmenting the dataset would enhance the precision of the model. Shill and Rahman (2021) reported that the plant disease detection performance of the YOLOv3 and YOLOv4 models achieved mean average precision (mAP) scores of 53% and 52%, respectively, and F1-scores of 55% and 56%. Despite training on a dataset from the PlantDoc repository in a laboratory setting, the YOLOv4 model demonstrated a superior mean average precision (mAP) compared to the YOLOv3 model.

Utilizing an improved YOLOv4 algorithm, Roy and Bhadiri (2021) implemented a real-time object identification system to identify diseases in apple plants. They sourced their dataset from Kaggle, which included 600 photographs for each of two categories depicting apple plant diseases. Following the implementation of several modifications to enhance its precision, the model underwent testing in a complex orchard environment. The technique obtained an F1-score of 95.9%, a mean average precision (mAP) value of 91.2%, and a frame rate of 56.9 frames per second (FPS). The created model and the original YOLOv4 model were analyzed to show that the former enhanced the mean average precision (mAP) by 9.05%, and the later improved the F1-score by 7.6%.

The BOOSTED-DEPICT model was initially proposed by Gokul Nath and Usha Devi (2020) for maize image clusterings. The approach combines regularized deep clustering techniques, deeply embedded clustering, and k-means clustering. On the Plant Village (PV) dataset, the model obtained a precision of 97.73%; on the PDD dataset, it obtained a precision of 91.25%. Using the YOLOv7 model, Nayar et al. (2022) found plant disease-es in the Plant Village dataset with a mean average precision (mAP) of 65%. Their findings, meanwhile, do not fit the criterion for instantaneous identification. Chen et al. (2022) created YOLOv5, a customized detection model designed to find powdery mildew on rubber trees. Training on a set comprising 2375 images, the model attained a mean average precision (mAP) of 70%. The upgraded version of YOLOv5 showed a 5.4% performance improvement compared to the previous version.

Akhalifiet et al. developed a transfer learning system with exceptional accuracy to effectively categorize pepper leaf disease. Three primary phases made up the study: classification; pre-processing (including data separation, augmentation, and resizing,); Pepper leaf diseases were precisely classified using MobileNet, ResNetV250, triple Fully Connected (FC) layers, VGG16 models. The experimental approach used a publicly available plant-village database of 1478 photos of healthy plants and 997 photographs of bacterial spots. Still, this method presents substantial challenges with data duplication during the training process.

3. Methodology

The subsections shown below can categorize the study's approach.

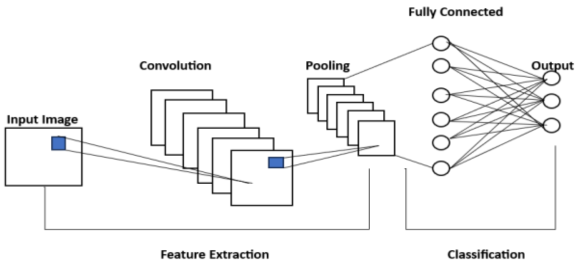


Figure 3 Architecture of CNN

3.1 Convolutional Neural Network

CNNs are highly effective in detecting and identifying pepper leaf diseases because they can extract hierarchical features from picture data. CNNs have demonstrated exceptional efficacy in various image classification and recognition endeavors, including identifying and categorizing diseases that impact pepper leaves, as shown in Figure 3. Here is a summary of how to use CNNs to detect and identify diseases that affect pepper leaves:

Data Collection and Preprocessing: We gather an extensive dataset of pepper leaf pictures, including a range of illnesses and healthy leaves. Prior to analysis, the photos undergo pre-processing to normalize their size, format, and color balancing, thereby ensuring consistency and improving the model's learning capabilities.

CNN Architecture: A CNN architecture is a specialised design that usually includes convolutional layers, pooling layers, and fully connected layers. These three types of layers are commonly used in neural networks. Convolutional layers extract features from the input images, while pooling layers reduce the dimensionality of these features. Finally, the fully associated layers use the extracted attributes to identify the input photographs.

Training Process: The CNN model is trained using pre-processed pictures of pepper leaves. During the training phase, the model develops the capacity to make connections between distinct characteristics and various categories of diseases. The parameters of the model are optimized to reduce the classification error. This procedure entails enhancing the model's performance by iteratively providing it with training data and fine-tuning its parameters to improve its precision. Utilize the pre-processed pepper leaf images to train the CNN model. Utilise a loss function, like cross-entropy loss, to measure the model's error and adjust its parameters using an optimizer, such as Adam.

Evaluation and Deployment: The CNN model that has undergone training is assessed by utilizing a distinct test dataset to determine its accuracy and capacity to generalize. Once the model has demonstrated satisfactory performance, real-world applications can implement it for practical use. This evaluation measures the model's precision and capacity to apply its knowledge to new, unfamiliar images, guaranteeing its effectiveness in detecting diseases. Real-world applications can use the model once it demonstrates adequate performance. Evaluate accuracy and generalization capability of the trained CNN model's accuracy and generalization capability by evaluating its performance on an independent test dataset. Compute measures such as accuracy, precision, and recall.

3.2 YOLOv7 (You Only Look Once, Version 7)

YOLOv7 is a cutting-edge object detection algorithm that has shown outstanding performance in several applications, such as detecting and identifying pepper leaf diseases. Figure 4 illustrates how the system's efficient structure and ability to identify multiple items in real-time make it an ideal choice for this undertaking. Below is a summary of how using YOLOv7 enables pepper leaf disease detection and identification:

Data Preparation: We have gathered a complete dataset of pepper leaf photos depicting various illnesses and healthy leaves. The photos are labeled with bounding boxes, indicating the affected regions or individual pepper leaves. These annotations provide the accurate and reliable information that the model uses to acquire knowledge. Acquire a dataset consisting of photographs of pepper leaves, accompanied by bounding boxes that accurately enclose the regions affected by diseases. Per-form image preprocessing to standardize the photo's size, format, and color balance.

Network Architecture: Employ the YOLOv7 architecture, comprising a backbone network, neck network, and prediction head. Optimise performance by adjusting the number of layers and filters in each layer.

Model Training: Utilising the annotated pepper leaf images, we proceed to train the YOLOv7 model. The model learns to associate the extracted features with the corresponding bounding boxes, enabling it to detect and localize diseased areas or pepper leaves in new images. Train the YOLOv7 model on pre-processed pepper leaf images with corresponding bounding boxes. Measure the model's error using a loss function, such as a combination of classification loss, localization loss, and confidence loss, and adjust the model's parameters using an optimizer, such as SGD or Adam.

Evaluation: Evaluate the accuracy and generalizability of the trained YOLOv7 model on a different dataset. Calculate various metrics, including average precision (AP), precision, recall, and mean average precision (mAP).

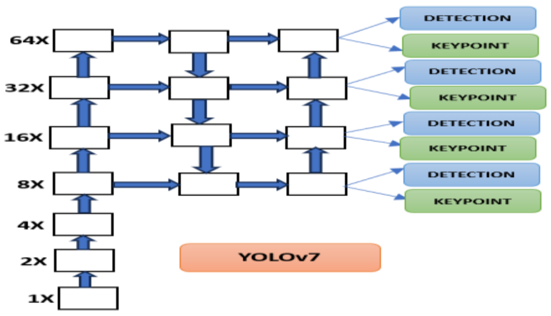


Figure 4 Architecture of YOLOv7

Inference and Disease Classification: After training, we can use the YOLOv7 model for real-time detection and recognition. Upon receiving a novel image of a pep-per leaf, the model promptly detects and precisely determines the location of any unhealthy regions or pepper leaves. Furthermore, it accurately categorizes them into the appropriate disease classifications.

3.3 Training labelled images using YOLOv7

YOLOv7, the latest iteration of the YOLO series, outperforms its previous versions in terms of accuracy, speed, and effectiveness, rendering it a potent instrument for identifying plant diseases. The model's compact dimensions, smaller than YOLOv7, make it ideal for deployment on resource-constrained devices like mobile phones and drones. This enables the immediate identification of diseases in outdoor settings. Un-like YOLOv7, YOLOv7 has superior detection accuracy, making it well-suited for situations where exact disease identification is crucial. The YOLOv7 architecture comprises four versions (YOLOv7s, YOLOv7m, YOLOv7l, and YOLOv7x) that offer distinct configurations to cater to various requirements and resource constraints. Users have the freedom to select the model that most closely aligns with their individual needs. YOLOv7 offers substantial benefits over prior iterations of YOLO and other object identification models in terms of accuracy, speed, and lightweight architecture, making it an excellent choice for plant disease diagnosis. The tool's capacity to provide accurate and prompt outcomes in contexts with limited resources renders it a useful asset for improving crop vitality, minimizing losses, and contributing to a more sustainable agricultural future.

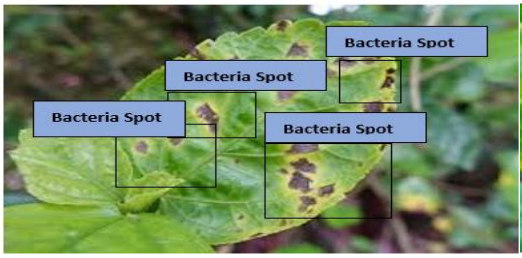
3.4 Training a custom YOLOv7 model

The procedure for training a personalized YOLOv7 model entails the following consecutive stages:

Configuring the YOLO setting: Retrieve the YOLOv7 repository from GitHub via the cloning process. If you have Kaggle or Colab installed, you should be able to exe-cute YOLOv7 on Torch. The PC will generate a new directory named "YOLOv7." This folder includes the weights of the pre-trained model and the well-organized YOLO directory structure.

Configuring the directory structure and data: To create the data folder, simply ensure it is located at the same level as the YOLO folder. Restrict access to the Images and Labels folders for the data folder. In the "Images and Labels" section, create the Train and Test directories. Uploading the labels to the data/labels/test and data/labels/train directories is essential. The labels file must have the identical name as the picture file, appended with addition of the ".txt" extension. When enumerating bounding boxes, each line should consist of exactly one bounding box. The subsequent enumeration comprises a comprehensive inventory of all items currently enclosed within the box in question. We use the starting value to indicate the class number. If there is only a singular class, the corresponding value is zero. The enclosing box's central pixel, normalized according to its width, takes second place. The third position indicates the normalized vertical location of the central pixel within the bounding box. Traditionally, the dimensions of the bounding box are often described in terms of its width and height. In order achieve consistency, we calculate the ratio of the number of pixels to the overall pixel count of the image. The normalized coordinates for a bounding box with pixel coordinates (20, 30) and dimensions of 50×60 on an image with dimensions (100, 100) are (0.2, 0.3, 0.5, 0.6). The quantity of label files and the quantity of lines within each label file have a direct correlation with the number of images. The following lines represent the number of bounding boxes contained in each image.

Configuring the configuration files in YAML: For the YOLOv7 model training procedure to begin, two YAML files must be used. The initial YAML file includes further details: The information required includes the coordinates of the training and test data, a count of classes (i.e., the different categories of objects to be detected), and the labels of the items belonging to those classes. The second YAML file includes the YOLOv7 backbone, YOLOv7 head, parameters, and anchor boxes.



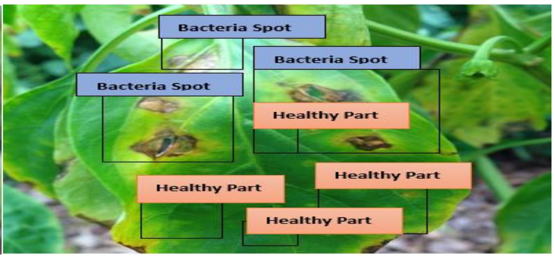


Figure 5 BS found in test dataset   Figure 6 BS and HP unaffected region

Training the model: To train the model, the notebook executes the train.py script. Users have the ability to define hyperparameters such as image dimensions, epoch count, and batch magnitude. A subdirectory stores the weights for the YOLOv7 model. The detect.py script enables leaf disease detection. As shown in Table 1, YOLOv7 creates a subdirectory after successfully completing the training procedure. You can denote the subfolder's directory as YOLOv7/run/train/exp.no./weights/last.pt. The dimension of the weight is reduced to 14.4 MB when YOLOv7s are utilized. YAML documentation. The YAML file we use will determine how to modify the weight file's size. Once the training is complete, we will produce the subsequent figures. Figures 5 and 6 display the findings obtained from the test data collected from the test dataset, whereas the last figure 7 represents the output of a leaf that has been randomly infected. Each iteration of YOLOv7 uses a data processor to process the training data, which then undergoes real-time enhancements. The enhancements include resizing, adjustments to color representation, and the creation of mosaic patterns. Mosaic data augmentation stands out as the most cutting-edge approach among these techniques.

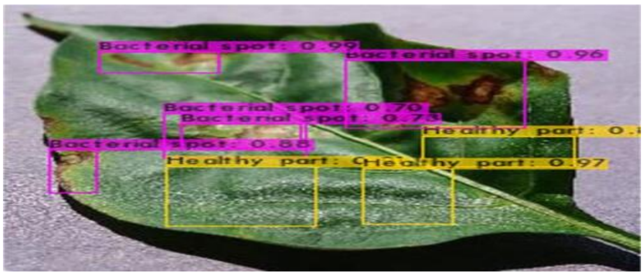


Figure 7 It shows a bacterial spot and an unaffected region identified by a Google image.

4. Proposed Architecture Framework

Figure 8 below illustrates the proposed model steps.

Image augmentation: DL must achieve great performance in categorizing diverse image types because of the need for a significant volume of data. Our model employs data augmentation to mitigate the inadequacy of the obtained dataset. Our model employs data augmentation to tackle the challenges posed by limited data and uneven image classification. This is used for data augmentation to enhance the training set by applying various modifications to the existing data. To guarantee enough data for our models and improve their performance, that can utilize several augmentation approaches to create numerous images for each category. The augmentation techniques used included a 45-degree rotation, as well as altering the width and height within a 0.2 range. That can also implement horizontal flipping.

Image preprocessing: This refers to the process of altering dataset photos before training the model. To optimise the performance of the model, it is common to employ preprocessing procedures. These procedures aim to improve the quality of the photographs, transform them into a more suitable format, and eliminate any possible noise from the camera acquisition process. The primary objective of imagine pre-processing is to eliminate undesirable distortions from the image, while preserving only the essential components necessary for analysis. This can employ a two-step approach to prepare the images, which involved reducing the image size and eliminating any background noise.

Image resizing: Prior to inputting the photos into the models, we altered the size of the images in the dataset from 256 × 256 to 128 × 128 to decrease their dimensions. The objective of picture scaling is to minimise the amount of time and computational complexity, while also reducing the image resolution to 256 × 256 pixels.

Noise removal: Efficiently removing background noise in illnesses that affect pep-per leaves and fruit requires image reconstruction methods that have remarkable noise reduction capabilities. Artefacts can arise at any stage of image capture, thereby undermining the effectiveness of the model. Therefore, the reduction of noise is of utmost importance in image processing algorithms. Image processing utilizes several methods for noise reduction, such as Median and gaussian filtering. This work aimed to reduce noise by means of median filtering and identify pepper leaf disease existence.

Image segmentation: This is the process of partitioning an image into distinct sections, with each segment representing an object or semantic concept. Image segmentation is essential in plant disease diagnosis as it allows for the isolation of specific plant components or lesions, enabling further investigation. This facilitates accurate identification and quantification of affected regions, thereby enabling focused interventions and enhanced agricultural administration. Segmentation techniques frequently employed include thresholding, edge detection, and region expansion, each with distinct advantages and constraints. Furthermore, CNNs, which are a type of DL method, have become highly effective tools for segmenting images. They have demonstrated remarkable performance in detecting plant diseases across several tasks.

Feature extraction: Pepper leaf analysis entails extracting pertinent data from the pepper plant's leaves and characterizing them according to the visual characteristics of the image, such as the region and edge components. This can identify and categorize the illnesses affecting pepper leaves based on the following criteria: To classify illness-es in pepper leaves effectively, because of that it can employed the CNN and YOLOv7 models to extract deep characteristics, which are essential for this purpose. Those are used with a fully connected layer as the classifier to categorize photos of pepper leaf disease using a fully connected layer classifier.

Features were extracted from CNN's concatenation model and YOLOv7-based CNN: Analysing and extracting variables such as colour, texture, edges, and segmentation is a crucial part of feature extraction. The input dataset consists of extracted data from image attributes. This study demonstrates the integration of the Convolutional Neural Network (CNN) and YOLOv7 by means of concatenation. The use of a hybrid model, which combines two distinct models, has significantly im-proved the results' precision. That can be created a hybrid technique to improve the precision of pepper leaf disease categorization. After combining these models, those can assess their performance by employing the fully connected layer classifier for training and model validation.

Classification: Pepper Leaf Classification: A Crucial Step for Disease Detection. Correctly classifying pepper leaves is a critical process for accurately identifying and controlling plant diseases. The procedure entails examining leaf photographs to differentiate between healthy leaves and unhealthy ones. That can use diverse methodologies for categorization, each with unique benefits and constraints:

1. Traditional Image Processing: This method employs algorithms to examine the appearances of leaves, such as color, texture, and form. Although it is uncomplicated and effective, it may encounter difficulties in detecting nuanced disease symptoms or intricate disease patterns.

2. ML (machine learning): That can train methods like SVMs and Random Forests using labelled datasets that contain both healthy and damaged leaves. These models can attain exceptional precision but necessitate a substantial amount of data and meticulous parameter adjustment.

3. Deep Learning: Convolutional Neural Networks (CNNs) excel at acquiring intricate patterns from photos through deep learning. The YOLOv7 model, a robust deep learning architecture, demonstrates exceptional performance in detecting and classifying objects, especially in the domain of detecting diseases affecting pepper leaves.

4. Hybrid Approaches: By integrating conventional methods with machine learning techniques, it is possible to capitalize on the advantages offered by both approaches. Using conventional methods to preprocess photos can strengthen machine learning models by emphasizing relevant features, resulting in enhanced categorization accuracy.

4.1 Algorithm

Dataset Collection (D): Collect a dataset D consisting of N images $\{\text{Image}_1, \text{Image}_2, \ldots, \text{Image}_N\}$and their corresponding labels $\{\boldsymbol{l_1}, \boldsymbol{l_2}, \ldots, \boldsymbol{l_N}\}$ Preprocess the images to standardize size and format:

$$\boldsymbol{D} = \{(\text{Image}_i, \boldsymbol{l_i})\}_{\boldsymbol{i=1}}^{\boldsymbol{N}} \qquad (1)$$

Image Processing (IP): Preprocess each image Image_i to ensure consistency and enhance features:

$$\text{Image}_i' = \text{IP}(\text{Image}_i) \qquad (2)$$

Noise Removal (NR): Apply a noise removal algorithm to clean images from unwanted artifacts:

$\text{Image}_i'=\text{NR}(\text{Image}_i)$ (3)

Segmentation (S): Segment each pre-processed image Imagei' into regions of interest (ROIs) representing individual leaves:

$\text{ROIs}_\text{i}=\text{S}(\text{Image}_\text{i}')$ (4)

Feature Extraction (FE): Extract features Features$_i$ from each ROI using a feature extraction method:

$\text{Features}_\text{i}=\text{FE}(\text{ROIs}_\text{i})$ (5)

Model Selection (MS): Choose the YOLOv7 architecture as the base model for pepper leaf disease detection:

$\text{Model} = \text{YOLOv7}$ (6)

Transfer Learning (TL): Initialize the YOLOv7 model with pre-trained weights θpre on a large dataset:

$\text{Model}_{\text{pre}} = \boldsymbol{YOLOv\ 7}\left(\boldsymbol{\theta}_{\text{pre}}\right)$ (7)

Data Augmentation (DA): Augment the training dataset D to increase diversity and robustness:

$\boldsymbol{D}_{\text{aug}} = \boldsymbol{Augmen\,t}(\boldsymbol{D})$ (8)

Hyperparameter Tuning (HT): Optimise hyperparameters Θ, including the learning rate, batch size, and choice of optimizer:

$\boldsymbol{\Theta}^* = \boldsymbol{arg}\ \underset{\boldsymbol{\Theta}}{\boldsymbol{min}}\boldsymbol{Loss}\left(\boldsymbol{D}_{\text{aug}}, \boldsymbol{\Theta}\right)$ (9)

Loss Function Selection (LFS): Choose a suitable loss function Loss to measure the discrepancy between predicted and true labels:

Localization Loss (Loc_loss): The localization loss evaluates the difference between the predicted and actual bounding box coordinates (bx, by, bw, bh) for each object. It is commonly calculated using a smooth L1 loss or a comparable regression loss function:

$$\text{Loc\_loss}= \sum_{\boldsymbol{i=1}}^{\boldsymbol{S^2}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{B}} \boldsymbol{1}_{\boldsymbol{ij}}^{\text{obj}} \left(\boldsymbol{\lambda}_{\text{coord}} \left[(\boldsymbol{x_i} - \hat{\boldsymbol{x}}_{\boldsymbol{i}})^{\boldsymbol{2}} + (\boldsymbol{y_i} - \hat{\boldsymbol{y}}_{\boldsymbol{i}})^{\boldsymbol{2}}\right] + \boldsymbol{\lambda}_{\text{coord}} \left[\left(\sqrt{\boldsymbol{w_i}} - \sqrt{\hat{\boldsymbol{w}}_{\boldsymbol{i}}}\right)^{\boldsymbol{2}} + \left(\sqrt{\left(\sqrt{\boldsymbol{h_i}} - \sqrt{\hat{\boldsymbol{h}}_{\boldsymbol{i}}}\right)^{\boldsymbol{2}}}\right)\right]\right) \quad (10)$$

Where: S_2 is the number of grid cells.

B is the number of bounding boxes per grid cell.

1_{ij}, 1^{obj} is an indicator function that equals 1 if object $_j$ is present in grid cell i, 0 otherwise.

$\boldsymbol{x_i}, \boldsymbol{y_i}, \boldsymbol{w_i}\boldsymbol{h_i}$ are the predicted bounding box coordinates and dimensions for object$_j$ in grid cell

i.

$\hat{\boldsymbol{x}}_{\boldsymbol{i}}, \hat{\boldsymbol{y}}_{\boldsymbol{i}}, \hat{\boldsymbol{w}}_{\boldsymbol{i}}, \hat{\boldsymbol{h}}_{\boldsymbol{i}}$ are the corresponding ground truth bounding box coordinates and dimensions.

λ_{coord} is a coefficient to balance the contribution of the bounding box coordinates to the overall loss.

Classification Loss (Cls_loss): The classification loss quantifies the difference between the predicted class probabilities and the ground truth class labels for each object. The SoftMax cross-entropy loss function is commonly used for this calculation:

$$\text{Cls\_loss} = -\frac{\boldsymbol{1}}{\boldsymbol{N}_{\text{obj}}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{S^2}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{C}}\boldsymbol{1}_{\boldsymbol{ij}}^{\boldsymbol{obj}}\sum_{\boldsymbol{c=1}}^{\boldsymbol{C}}\left[\boldsymbol{p_i(c)log}\left(\hat{\boldsymbol{p}}_{\boldsymbol{i}}(\boldsymbol{c})\right)+(\boldsymbol{1}-\boldsymbol{p_i(c)})\boldsymbol{log}\left(\boldsymbol{1}-\hat{\boldsymbol{p}}_{\boldsymbol{i}}(\boldsymbol{c})\right)\right]$$
(11)

Where: N_{obj} is the total number of objects. C is the total number of classes.

$\boldsymbol{p_i(c)}$ is the predicted probability of object $_j$ belonging to class c.

$\hat{\boldsymbol{p}}_{\boldsymbol{i}}(\boldsymbol{c})$ is the ground truth probability (1 if object $_j$ belongs to class c, 0 otherwise).

Total Loss: The total loss is calculated by combining the localization loss and the classification loss using a weighted sum.

$$\text{Total loss} = \text{Loc\_loss} + \boldsymbol{\lambda_{cls}} \times \text{Cls\_loss} \qquad (12)$$

Where: λcls is a coefficient to balance the contribution of the classification loss to the overall loss. Choosing appropriate values for λcoord and λcls is crucial to ensure a balanced contribution of localization and classification losses to the overall loss function. These coefficients are typically chosen empirically or through validation experiments.

Model Evaluation and Validation (MEV): Assess the performance of the trained model using a separate validation dataset to verify its ability to generalise.

$$\text{Performance=Evaluate (Model, } D_{\text{val}}) \qquad (13)$$

5. Experimental Results and Discussions

5.1 Plant Village Dataset for CNN And YoloV7

To optimize CNN-YOLOv7 models for pepper leaf disease detection and identification, acquiring a comprehensive dataset is crucial. For this purpose, the plant village dataset is a valuable resource. It comprises high-resolution images of various plant diseases, including those affecting pepper leaves. The dataset provides diverse examples of healthy and diseased leaves, enabling the model to learn the distinguishing features effectively. Labels indicate the type and severity of the disease present in each image. This rich annotation facilitates precise training of the CNN-YOLOv7 model, allowing it to accurately identify and classify pepper leaf diseases in real-time. By leveraging this dataset, researchers can enhance the model's performance, leading to more reliable detection and timely intervention to mitigate crop losses.

The Plant Village Dataset comprises 50,000 images depicting both healthy and dis-eased plant leaves, encompassing 38 different species and 14 distinct disease categories. Training DL models to detect and categorize plant diseases is an excellent meth-od. Both CNNs (image classification) and YOLOv7 (object detection and localization) can use the dataset for object detection tasks. However, it has some problems, such as differences in picture quality, a lack of geographic diversity, and dataset imbalances. There are some problems with the Plant Village Dataset, but its size, diversity, and annotations make it a useful resource for scientists and coders working in this area. Convolutional Neural Networks (CNN) and YOLOv7 likely use the Plant Village dataset, which likely contains a variety of images labeled with different plant species, diseases, and growth stages. The dataset is designed to facilitate the training and evaluation of deep learning models, specifically CNNs and YOLOv7, with the goal of accurately detecting and categorising plant diseases. There are bounding boxes, or segmentation masks, on each picture in the dataset that show where and how bad plant diseases are. This large dataset makes it possible to build strong models that can find and classify plant diseases in the real world. This makes it useful for agricultural studies and precision farming. Researchers and professionals can use this dataset to improve how well CNNs and YOLOv7 do at automatically diagnosing and keeping focused on plant health. Table 3 displays existing comparisons, while Fig 8 presents a comparison of the suggested methodology.

Table 1 Quantity of photos for the test and train datasets.

Class No.	Class Name	Training Samples	Testing Samples	Total Samples
0	Healthy Part	1000-2000	200-500	2500
1	Bacterial leaf spot	1000-2000	200-500	2500
2	Mosaic virus	1000-2000	200-500	2500
3	Blight	1000-2000	200-500	2500
4	Damping off disease	1000-2000	200-500	2500
5	Verticillium wilt	1000-2000	200-500	2500
6	Pests	1000-2000	200-500	2500
7	Blossom end rot	1000-2000	200-500	2500
8	Yellowing leaves	1000-2000	200-500	2500
9	Curling leaves	1000-2000	200-500	2500

The experimental results, based on CNN and YOLOv7, were obtained prior to the proposed hybrid model before pre-processing: In this stage, our preprocessing first tested the proposed model's efficacy without applying any noise reduction, dataset segmentation, or augmentation methods. The dataset must be reduced in size from 256 x 256 to 128 × 128 before it can be entered into the suggested model. As demonstrated in Figure 11 below, resizing is crucial for achieving optimal results in training, validation, testing, loss, and model performance.

Experimental results after noise elimination for the proposed hybrid model based on CNN and YOLOv7: The noise removal investigations use Gaussian filters as the filtering method. Three models utilize this noise reduction strategy: the CNN model, YOLOv7, and the hybrid future model post-concatenation. Figure 13 displays the performance graph for the classifiers Resnet50, VGG16, and InceptionV3, and com-pares them with the experiment in Table 5. The proposed concatenated model first removed noise before executing picture segmentation. The experiment resulted in a training accuracy of 86.78%, a validation accuracy of 83.05%, and a testing accuracy of 81.76%. Initially training the model with the concatenation technique improves the classification accuracy of the concatenated model.

Experimental results following hybrid suggested concatenated model segmentation based on CNN and YOLOv7: In terms of threshold segmentation, our hybrid model used the threshold method to segment our picture datasets.

Our proposed methodology includes adaptive thresholding in our hybrid model. This technique typically requires an input image that has been filtered using a Gaussian filter, and it produces a binary image that represents the segmentation. Our hybrid model assigns the background value to pixel values below the threshold and assumes the foreground value for values equal to or over the threshold. After implementing adaptive segmentation, our hybrid model continued to evaluate the experiment's outcomes. However, our hybrid model will provide you with specific examples of previous model evaluations that employed the PlantVillage dataset, as illustrated in Table 2-7 and Figure 9-14.

Table 2 Existing Performance and Evaluation of Resnet50, VGG16 and Inceptionv3 Classifiers.

Classifier	Performance Metrics			
	Accuracy	Loss	F1-Score	AUC
Vgg16	99.73%	0.0117	0.9978	0.997
ResNet50	99.32%	0.0166	0.9943	0.995
InceptionV3	95.7%	0.1241	0.9686	0.954

Figure 9 Performance graph for VGG16, ResNet50 and InceptionV3 classifier

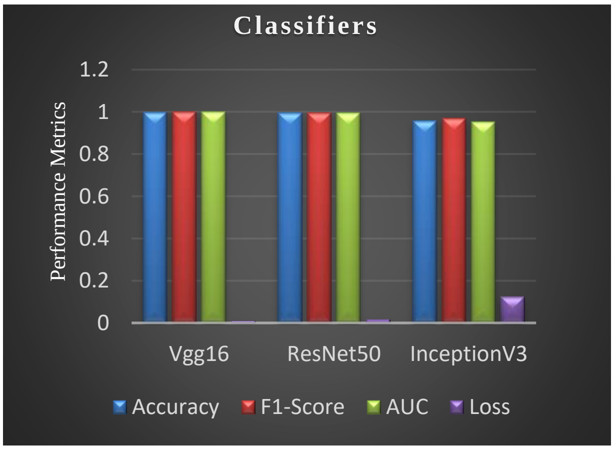


Table 3 Model Evaluation Using the PlantVillage Dataset with the Proposed Model(Alhashmi et al. 2024).

Models	Training		Validation	
	Accuracy	Loss	Accuracy	Loss
DenseNet-121	98.03	0.04	96.76	0.11
NASNetLarge	97.77	0.07	97.03	0.11
DenseNet-201	98.15	0.03	98.68	0.05
MobileNet-V2	98.34	0.06	94.68	0.17
ResNet152-V2	98.36	0.06	93.03	0.24
VGG-19	98.27	0.04	97.73	0.08

Xception	97.87	0.07	94.29	0.22
EfficientNet-B5	96.60	0.11	88.45	0.43
EfficientNet-B7	96.21	0.11	93.98	0.18
Hybrid-model (EfficientNetB7 + ResNet152V2)	97.81	0.06	94.00	0.21
Proposed Approach (CNN+YOLOv7)	98.92	0.06	99.02	0.08

Figure 10 Evaluation Graph Using the PlantVillage Dataset with the Proposed Model

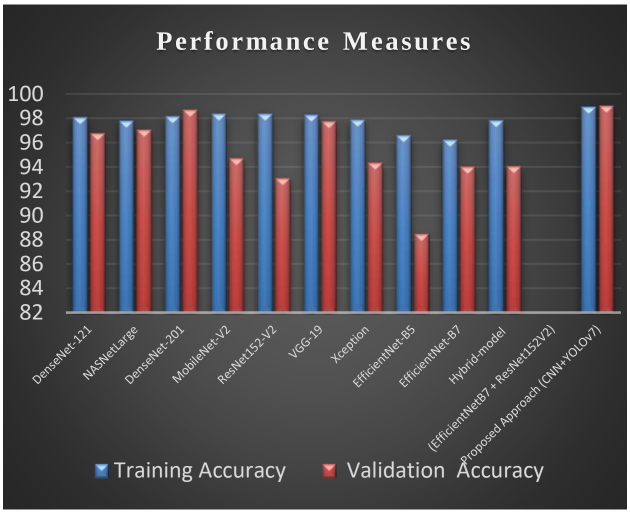


Table 4 Model performance for various classes of the PlantVillage dataset with the proposed model

Models	Class	Training		Validation	
		Accuracy	Loss	Accuracy	Loss
DenseNet-12	Pepper bell bacterial spot	86.27	0.595	98.76	0.952
NASNetLarge		98.41	0.577	94.58	0.355
DenseNet-201		97.89	0.561	98.57	0.258
MobileNet-V2		91.31	0.468	93.49	0.255
ResNet152-V2		90.87	0.482	91.57	0.201
VGG-19		90.85	0.869	98.61	0.235
Xception		98.83	0.731	98.67	0.952
EfficientNet-B5		94.77	0.81	98.55	0.28
EfficientNet-B7		91.15	0.84	94.58	0.47
Hybrid-model (EfficientNetB7 + ResNet152V2)		96.34	0.15	91.47	0.58
Proposed Approach(CNN+YOLOv7)		98.92	0.06	99.02	0.08

Figure 11 Evaluation Graph Using the PlantVillage Dataset on Various Classes with the Proposed Model

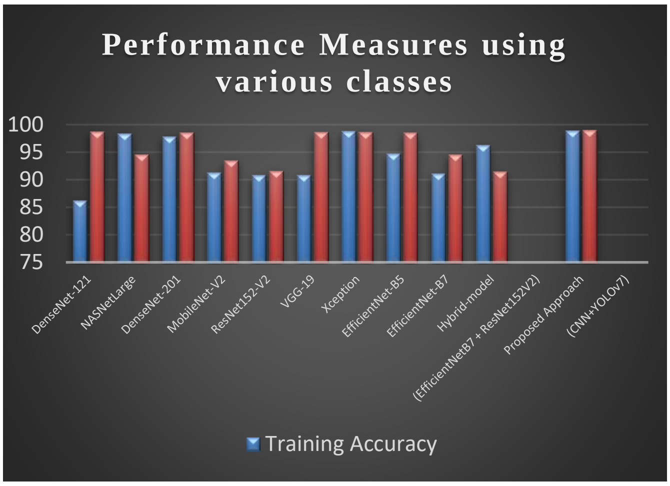


Table 5 Model performance for various classes of the PlantVillage dataset with the Proposed Model

Models	Class	Training		Validation	
		Accuracy	Loss	Accuracy	Loss
DenseNet-121	Pepper bell healthy	98.85	0.74	98.97	0.74
NASNetLarge		94.77	0.81	96.01	0.81
DenseNet-201		91.13	0.82	80.60	0.591
MobileNet-V2		96.34	0.13	98.74	0.009
ResNet152-V2		97.36	0.48	98.53	0.023
VGG-19		96.76	0.865	98.74	0.009
Xception		94.94	0.597	98.95	0.002
EfficientNet-B5		97.84	0.751	80.62	0.591
EfficientNet-B7		96.33	0.588	98.23	0.58
Hybrid-model (EfficientNetB7 + ResNet152V2)		91.48	0.862	94.45	0.47
Proposed Approach (CNN+YOLOv7)		98.92	0.06	99.02	0.08

Figure 12 Evaluation Graph Using the PlantVillage Dataset on Various Classes with the Proposed Model

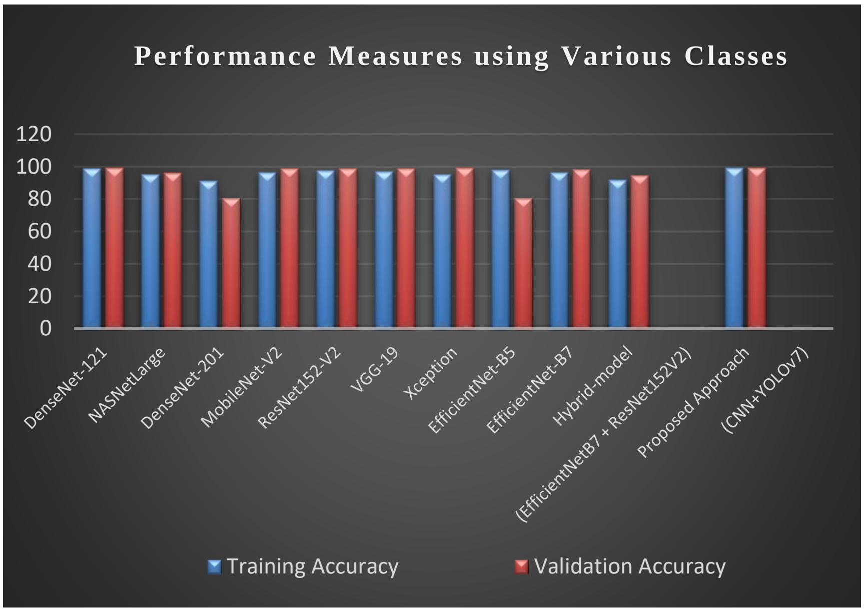


Table 6 shows the comparative analysis provides valuable insights into the performance of different machine learning models on various metrics. By understanding the strengths and weaknesses of each model.

Table 6 Performance Metrics for Exiting Model to Proposed Model

Model	Accuracy	Precision	Recall	F1 Score	Specificity
DenseNet-121	85.3	85.9	86.7	86.9	87.2
NASNetLarge	87	87.2	87.9	88.2	88.7
DenseNet-201	88.6	88.9	89.1	89.3	89.6
MobileNet-V2	89.5	89.7	90.3	90.5	90.8
ResNet152-V2	90.9	91.3	91.7	92.0	92.5
VGG-19	93.1	93.5	93.9	94.2	94.7
Xception	94.5	94.8	95.1	95.3	95.9
EfficientNet-B5	95.2	95.4	95.6	96.2	96.3
EfficientNet-B7	96.1	96.4	96.7	97.2	97.0
Hybrid-model (EfficientNetB7 + ResNet152V2)	97.0	97.4	97.6	98.1	98.4
ProposedApproach (CNN+YOLOv7)	98.9	98.5	98.0	99.02	99.1

In Fig 13. presents a comparative analysis of various machine learning models based on different performance metrics. These metrics are used to evaluate the effectiveness of each model in a specific task.

Figure 13 Evaluation Graph on Various Existing model with the Proposed Model

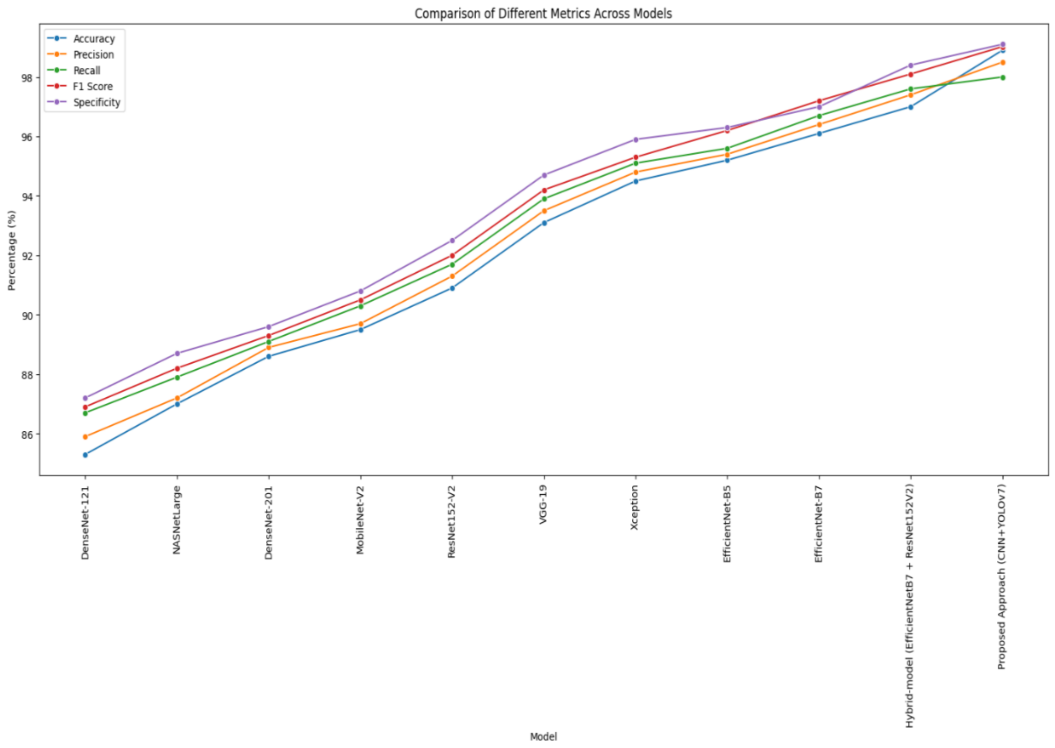


Table 7 Comparison of state-of-the-art testing accuracies and training parameters with the suggested work

Ref. and year	Number of Class and Plant	Methods	Accuracy (%)
Proposed Approach	9(C)-1 (P)	CNN-YOLOv7	98.76
(SERT 2021)	5(C)-2 (P)	FRCNN and Google Net	98.08
(Zhou et al. 2019)	3(C)-1 (P)	FCM-KM algorithm and FRCNN fusion.	97.4
(Ozguven and Adem 2019)	4(C)-1 (P)	FRCNN	95.5
(Sibiya and Sumbwanyambe 2019)	4(C)-1 (P)	CNN	94.0
(Hu et al. 2019)	4(C)-1 (P)	CIFAR10 dataset-quick (Convolutional Neural Network)	92.6
(Bhagat and Kumar 2022)	2 class for 1 type Plant	(CNN) Convolutional Neural Network	96.78
(Francis, Anto Sahaya Dhas D, and Anoop B K 2016b)	2 class for 1 type Plant	SOFT COMPUTING	-
(Alagumariappan et al. 2020)	-	Machine Learning	95
(Mathew and Mahesh 2022)	2 class for 1 type Plant	YOLOv5	95.02
(Bezabh et al. 2023)	2 class for 1 type Plant	CNN	97.82
(Boukabouya, Moussaoui, and Berrimi 2022)	-	Vision Transformers (ViT)	96.7, 98.52, 99.1 and 99.7
(Ahmed, Ahad, and Emon 2023)	1 class for 1 type plant	Machine Learning	-

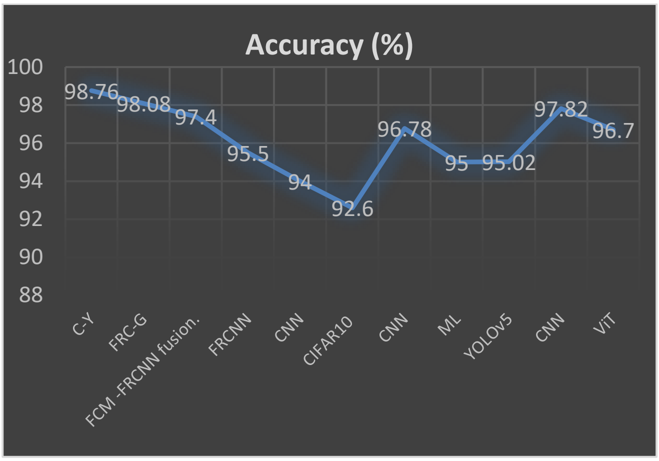


Figure 14. Comparative of the proposed approach

6. Conclusion

In this paper, our hybrid model tested a concatenation-based CNN and YOLOv7 model for pepper leaf disease detection. The CNN-YOLOv7 model outperformed the CNN and YOLOv7 models. CNN accurately classified healthy and diseased leaves, while YOLOv7 discovered unhealthy areas. Both models showed improved accuracy after noise removal, demonstrating their tolerance to noise interference. The CNN-YOLOv7 approach has significant potential for accurate and quick pepper leaf dis-ease diagnosis, which aids precision agriculture and disease management. Noise removal improved both models' performance, and YOLOv7 can identify and localize many pathologies. The model had an average precision of 98.92% and 99.02%, providing real-time field-based illness assessment. If adopted, the CNN-YOLOv7 model could manage pepper sickness, promoting sustainable agriculture and food security.

7. Further Scope

The potential for optimizing CNN-YOLOv7 models in the context of pepper leaf dis-ease detection and identification is extensive and diverse in nature. To begin with, future investigations may focus on integrating supplementary data augmentation methodologies in order to strengthen the model's resilience and applicability to a wide range of environmental circumstances and leaf aesthetic variations. Also, looking into whether transfer learning is possible using datasets about similar plant diseases could allow CNN-YOLOv7 models to be changed to find diseases in more goods, which would increase the framework's usefulness. Also, looking into how to use edge computing solutions that allow real-time inference on devices that are out in the field would make the optimized models much more useful in real life. This would empower farmers to make prompt interventions and decisions. Furthermore, the integration of explainability techniques, which provide interpretable insights into the model's decision-making process, could enhance trust and adoption among end-users, thereby facilitating the seamless integration of agricultural operations. In conclusion, investigating collaborative methodologies that incorporate multidisciplinary groups consisting of engineers, computer scientists, and agronomists may promote innovation and stream-line the creation of comprehensive resolutions to the intricate obstacles linked to the detection and control of plant diseases in agricultural environments.

References

1. Ahmed, Faruk, Md. Taimur Ahad, and Yousuf Rayhan Emon. 2023. "Machine Learning-Based Tea Leaf Disease Detection: A Comprehensive Review."
2. Alagumariappan, Paramasivam, Najumnissa Jamal Dewan, Gughan Narasimhan Muthukrishnan, Bhaskar K. Bojji Raju, Ramzan Ali Arshad Bilal, and Vijayalakshmi Sankaran. 2020. "Intelligent Plant Disease Identification System Using Machine Learning." P. 49 in 7th International Electronic Conference on Sensors and Applications. Basel Switzerland: MDPI.
3. Alhashmi, Asma A., Manal Abdullah Alohali, Nazir Ahmad Ijaz, Alaa O. Khadidos, Omar Alghushairy, and Ahmed Sayed. 2024. "Bayesian Optimization with Deep Learning Based Pepper Leaf Disease Detection for Decision-Making in the Agricultural Sector." AIMS Mathematics 9(7):16826–47. doi: 10.3934/math.2024816.
4. Bezabh, Yohannes Agegnehu, Ayodeji Olalekan Salau, Biniyam Mulugeta Abuhayi, Abdela Ahmed Mussa, and Aleka Melese Ayalew. 2023. "CPD-CCNN: Classification of Pepper Disease Using a Concatenation of Convolutional Neural Network Models." Scientific Reports 13(1):15581. doi: 10.1038/s41598-023-42843-2.
5. Bhagat, Monu, and Dilip Kumar. 2022. "A Comprehensive Survey on Leaf Disease Identification &amp; Classification." Multimedia Tools and Applications 81(23):33897–925. doi: 10.1007/s11042-022-12984-z.
6. Boukabouya, Rayene Amina, Abdelouahab Moussaoui, and Mohamed Berrimi. 2022. "Vision Transformer Based Models for Plant Disease Detection and Diagnosis." Pp. 1–6 in 2022 5th International Symposium on Informatics and its Applications (ISIA). IEEE.
7. Camargo, A., and J. S. Smith. 2009. "An Image-Processing Based Algorithm to Automatically Identify Plant Disease Visual Symptoms." Biosystems Engineering 102(1):9–21. doi: 10.1016/j.biosystemseng.2008.09.030.
8. Francis, Jobin, Anto Sahaya Dhas D, and Anoop B K. 2016a. "Identification of Leaf Diseases in Pepper Plants Using Soft Computing Techniques." Pp. 168–73 in 2016 Conference on Emerging Devices and Smart Systems (ICEDSS). IEEE.
9. Francis, Jobin, Anto Sahaya Dhas D, and Anoop B K. 2016b. "Identification of Leaf Diseases in Pepper Plants Using Soft Computing Techniques." Pp. 168–73 in 2016 Conference on Emerging Devices and Smart Systems (ICEDSS). IEEE.
10. Hu, Gensheng, Xiaowei Yang, Yan Zhang, and Mingzhu Wan. 2019. "Identification of Tea Leaf Diseases by Using an Improved Deep Convolutional Neural Network." Sustainable Computing: Informatics and Systems 24:100353. doi: 10.1016/j.suscom.2019.100353.
11. Mathew, Midhun P., and Therese Yamuna Mahesh. 2022. "Leaf-Based Disease Detection in Bell Pepper Plant Using YOLO V5." Signal, Image and Video Processing 16(3):841–47. doi: 10.1007/s11760-021-02024-y.
12. Ozguven, Mehmet Metin, and Kemal Adem. 2019. "Automatic Detection and Classification of Leaf Spot Disease in Sugar Beet Using Deep Learning Algorithms." Physica A: Statistical Mechanics and Its Applications 535:122537. doi: 10.1016/j.physa.2019.122537.
13. Ramanjot, Usha Mittal, Ankita Wadhawan, Jimmy Singla, N. Z. Jhanjhi, Rania M. Ghoniem, Sayan Kumar Ray, and Abdelzahir Abdelmaboud. 2023. "Plant Disease Detection and

Classification: A Systematic Literature Review." Sensors 23(10):4769. doi: 10.3390/s23104769.

14. Revathi, P., and M. Hemalatha. 2012. "Classification of Cotton Leaf Spot Diseases Using Image Processing Edge Detection Techniques." Pp. 169–73 in 2012 International Conference on Emerging Trends in Science, Engineering and Technology (INCOSET). IEEE.
15. Rubia J, Jency, and Babitha Lincy R. 2021. "Detection of Plant Leaf Diseases Using Recent Progress in Deep Learning-Based Identification Techniques." ITEGAM- Journal of Engineering and Technology for Industrial Applications (ITEGAM-JETIA) 7(30). doi: 10.5935/jetia.v7i30.768.
16. Sankaran, Sindhuja, Ashish Mishra, Reza Ehsani, and Cristina Davis. 2010. "A Review of Advanced Techniques for Detecting Plant Diseases." Computers and Electronics in Agriculture 72(1):1–13. doi: 10.1016/j.compag.2010.02.007.
17. Sannakki, Sanjeev S., Vijay S. Rajpurohit, V. B. Nargund, and Pallavi Kulkarni. 2013. "&amp;#x201C;Diagnosis and Classification of Grape Leaf Diseases Using Neural Networks&amp;#x201D;" Pp. 1–5 in 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT). IEEE.
18. SERT, Eser. 2021. "A Deep Learning Based Approach for the Detection of Diseases in Pepper and Potato Leaves." ANADOLU JOURNAL OF AGRICULTURAL SCIENCES 167–78. doi: 10.7161/omuanajas.805152.
19. Sibiya, Malusi, and Mbuyu Sumbwanyambe. 2019. "A Computational Procedure for the Recognition and Classification of Maize Leaf Diseases Out of Healthy Leaves Using Convolutional Neural Networks." AgriEngineering 1(1):119–31. doi: 10.3390/agriengineering1010009.
20. Singh, Vijai, and A. K. Misra. 2017. "Detection of Plant Leaf Diseases Using Image Segmentation and Soft Computing Techniques." Information Processing in Agriculture 4(1):41–49. doi: 10.1016/j.inpa.2016.10.005.
21. Zhou, Guoxiong, Wenzhuo Zhang, Aibin Chen, Mingfang He, and Xueshuo Ma. 2019. "Rapid Detection of Rice Disease Based on FCM-KM and Faster R-CNN Fusion." IEEE Access 7:143190–206. doi: 10.1109/ACCESS.2019.2943454.