Received: 18 September 2024 | Revised: 9 June 2025 | Accepted: 11 July 2025 | Published online: 24 July 2025

Artificial Intelligence and Applications
2025, Vol. 3(4) 385-391
DOI: 10.47852/bonviewAIA52024376

BON VIEW PUBLISHING

RESEARCH ARTICLE

Integrating Pattern Recognition and CNN-Based Models for Improved Bean Disease Detection and Agricultural Yield Enhancement

Farian S. Ishengoma[1,*] and **Joseph P. Telemala**[1]

[1]*Department of Informatics and Information Technology, Sokoine University of Agriculture, Tanzania*

Abstract: Early detection of plant diseases is crucial for minimizing crop losses and improving agricultural productivity. However, current manual detection methods are time-consuming and inaccurate. Although advanced techniques such as pattern recognition (PR) and image processing have demonstrated potential in automating disease identification, there is still a need for more accurate and efficient methods that can handle various image qualities such as complex backgrounds and complex leaf structures. This paper introduces a method that uses PR and convolutional neural network (CNN) models to detect and classify diseases in infected bean leaves based on shape features. Using contour detection, a computer-vision-based PR method, we can effectively detect and distinguish different shapes in images. Two datasets were prepared: original bean leaf images subjected to background removal and enhanced images subjected to background removal, binary thresholding, and contour detection. These preprocessing steps reduced noise, improved color differentiation, and enhanced shape detection for more precise disease identification. Experimental results show that the models trained on the enhanced images showed up to an 8.16% improvement in accuracy, precision, and sensitivity compared to those trained on the original images. This study highlights the potential of integrating image processing techniques with CNN models for more accurate and efficient plant disease detection.

Keywords: beans, machine learning, convolutional neural network, pattern recognition, contour detection

1. Introduction

Plant diseases and pests pose a significant threat to crops, potentially causing severe damage if not identified early. These issues lead to reduced crop yields, resulting in substantial financial losses. Without early detection, crop losses can exceed 45% annually, even with the use of pesticides [1–5]. Automatic disease identification in agriculture is among the several applications that have emerged because of recent advancements in artificial intelligence (AI) and computer vision (CV). Plant disease detection systems have been developed using AI approaches, particularly deep learning [6, 7].

Several image-based studies have integrated AI, machine learning (ML), and pattern recognition (PR) to detect plant leaf diseases and pests. Singh et al. [8] proposed several optimization strategies and convolutional neural network (CNN)-based models for detecting diseases in a bean leaf image. The models were trained on a publicly available dataset consisting of 1295 images captured using a smartphone camera. The dataset has three classes, namely, angular leaf spot, bean rust, and healthy. They used various optimization strategies, including Adam, stochastic gradient descent (SGD), and Nadam, to train different CNN-based models. After training the models for 25 epochs, their experimental results showed that the EfficientNetB6 architecture with the Adam optimizer achieved the highest accuracy of 91.74%. However, the limited preprocessing techniques used led to reduced classification accuracy in the implemented model. Elfatimi et al. [9] introduced a method that uses MobileNet models to classify bean leaves into three classes, namely, angular leaf spot, bean rust, and healthy. They used a publicly available dataset with 1296 images. They compared and evaluated various architectures with different hyperparameters and optimization methods, employing five different optimizers, including Adagrad, Nadam, SGD, RMSprop, and Adam, with asynchronous gradient descent. After training for 100 epochs, the MobileNet model outperformed others, achieving a 92% accuracy on the testing data. However, its efficiency dropped by 5% during testing.

Sembiring et al. [10] used a CNN architecture, such as VGGNet, ShuffleNet, and SqueezeNet, to detect 10 diseases in tomato plants using leaf images. This method achieved an accuracy rate of 97.15% while maintaining model simplicity. Gui et al. [11] developed a method for the early diagnosis of soybean mosaic virus disease (SMV) using hyperspectral images, grading severity as 0, 1, or 2. Their hybrid model combining CNN and support vector machine (SVM) achieved accuracy rates of 96.67% for the training set and 94.17% for the test set. The model's efficiency dramatically dropped when the sample size was halved, but it maintained an accuracy above 90%, demonstrating robustness with fewer samples. Moreover, Agarwal et al. [12] introduced a CNN-based architecture for classifying tomato crop leaf diseases using a Plant-Village dataset that is publicly available. The dataset consists of 50,000 images with 14 classes. After training the models for 1000 epochs, their proposed method achieved an accuracy of 91.2%. However, this method was susceptible to overfitting, particularly with a limited number of classes. Aravind et al. [13] used a CNN, specifically AlexNet, to classify three diseases and healthy leaves of grape images from the Plant-Village dataset. Using a transfer learning approach with pretrained AlexNet, the model achieved a remarkable 97.62% classification accuracy. Further performance analysis using multiclass support vector machine (MSVM)

***Corresponding author:** Farian S. Ishengoma, Department of Informatics and Information Technology, Sokoine University of Agriculture, Tanzania. Email: farishen@sua.ac.tz

revealed that features from the ReLU 3 layer of AlexNet attained the highest classification accuracy of 99.23%. Picon et al. [14] developed a method for detecting three wheat diseases in natural environments using the ResNet50 model. The model was trained on 8,178 images captured with a mobile phone, achieving an accuracy of 96%.

Yang et al. [15] conducted research aimed at identifying abnormal hydroponic lettuce leaves, specifically yellow and rotten varieties, using multiple linear regression (MLR), K-nearest neighbor (KNN), and SVM techniques. MLR recorded detection accuracies of 89.48% for yellow leaves and 99.29% for rotten leaves, and SVM achieved 98.33% accuracy for yellow leaves and 97.91% for rotten ones. In contrast, KNN demonstrated a significant disadvantage with a longer detection time of 20.25 s, compared to 0.61 s for MLR and 1.98 s for SVM. Additionally, Pantazi et al. [16] introduced a method for categorizing various plant diseases by employing the local binary pattern algorithm in conjunction with an SVM classifier. Although this approach exhibited strong generalization capabilities, its classification performance diminished when faced with noisy samples, resulting in an average accuracy of 95%.

Previous studies have used ML- and CNN-based models to detect crop diseases in images, but these methods face challenges when extracting disease patterns from images with complex backgrounds. Issues such as varying lighting conditions, overlapping objects, and similarities between diseased plants and background elements complicate this task. Traditional ML algorithms struggle to consistently identify disease and pest-related elements in such dynamic environments, resulting in unstable models for disease and pest detection. Additionally, these algorithms often fail to generalize well with limited datasets, especially after only a few training epochs. Relying solely on optimization strategies to improve model generalization is insufficient.

To overcome these challenges, this study employs various preprocessing methods and PR techniques to enhance image quality in complex environments, ultimately improving the performance of ML models on limited datasets. The method integrates PR techniques with CNN-based models to enhance the detection of disease on bean leaves. These PR techniques play a crucial role in separating background from foreground images and locating infected areas, thereby facilitating CNN-based models in efficiently identifying the affected regions. The CNN-based models used in this study are trained on both original and enhanced datasets to compare their performance before and after implementing the proposed method. The original images undergo background removal to reduce background interference, whereas the enhanced images undergo additional processes, including background removal, binary thresholding, and contour detection. These combined strategies effectively minimize image noise and enhance the accuracy of disease diagnosis.

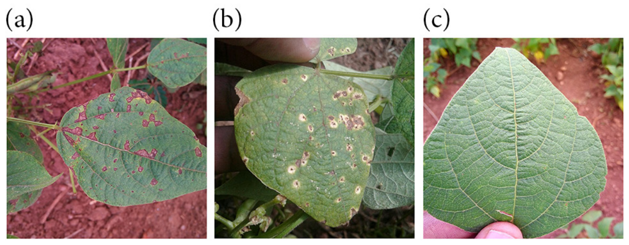
The remainder of this paper is structured as follows. The next section describes the materials and methodology, followed by Section 3, which covers the experimental setup. Section 4 discusses the experiment's results, and finally, the conclusion is provided in Section 5.

2. Materials and Method

2.1. Dataset

This study utilized a publicly available bean leaf disease dataset, which can be accessed from the Makerere AI Lab [17]. This dataset consists of 1,296 images, each with a resolution of 500 × 500 pixels, and is categorized into four groups: angular leaf spot (Als) with 432 images, bean rust (Br) with 436 images, and healthy (He) with 428 images. Figure 1 shows the three categories of the bean leaf images that were captured in a normal environment.

Figure 1
Bean leaf dataset: (a) angular leaf spot, (b) bean rust, and (c) healthy



The dataset used in this study contains a total of 4,040 images, divided in a 70:20:10 ratio for training, validation, and testing. To meet the requirement of 500 images for each group, the images were expanded using the 90-degree rotation method from the image augmentation technique, as shown in Figure 2. In this stage, we employed only one augmentation technique to obtain the minimal images required for each class because the Als, Br, and He classes required 68, 64, and 72 images, respectively. During model training, further augmentation approaches were used to improve the learning process and performance.

2.2. Preprocessing method

In the preprocessing stage, we employed three methods to minimize noise: background removal, binary thresholding, and contour detection, as depicted in Figure 3. The background was removed using Python's remove-background "rembg" function, which reduces distracting noise. This function utilizes a neural network model that has been trained on an extensive dataset to efficiently remove backgrounds from the images.

A binary threshold is employed to discern color variations in images. Every pixel in a binary image must have one of two values, i.e., 0 or 1, indicating black or white, respectively (see Equation (1)). The binary threshold process involves establishing a specific threshold value and subsequently assigning pixels in the input image to either 0 or 1. This assignment is based on whether the intensity value of each pixel is less than or greater than the threshold value [18].

$$B(x,y) = \begin{cases} 1 & \text{if } I(x,y) \geq T \\ 0 & \text{if } I(x,y) < T' \end{cases} \tag{1}$$

where T is the threshold value, I(x,y) is the intensity value, and B (x, y) is the binary image.

Once a binary image is obtained, contour (C) can be defined as a set of connected points where the binary value changes from 0 to 1 or vice versa, as shown in Equation (2).

$$C = \{(x,y) | B(x,y) = 1 \text{ and } \exists (x',y') \in N(x,y) \\ \text{such that } B(x',y') = 0\}, \tag{2}$$

where N(x,y) is the neighborhood of point (x, y).

Figure 2
Image augmentation technique used in this study: (a) original image and (b) 90-degree rotated image

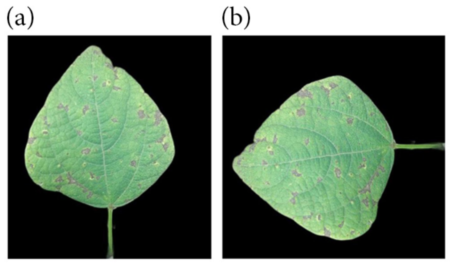
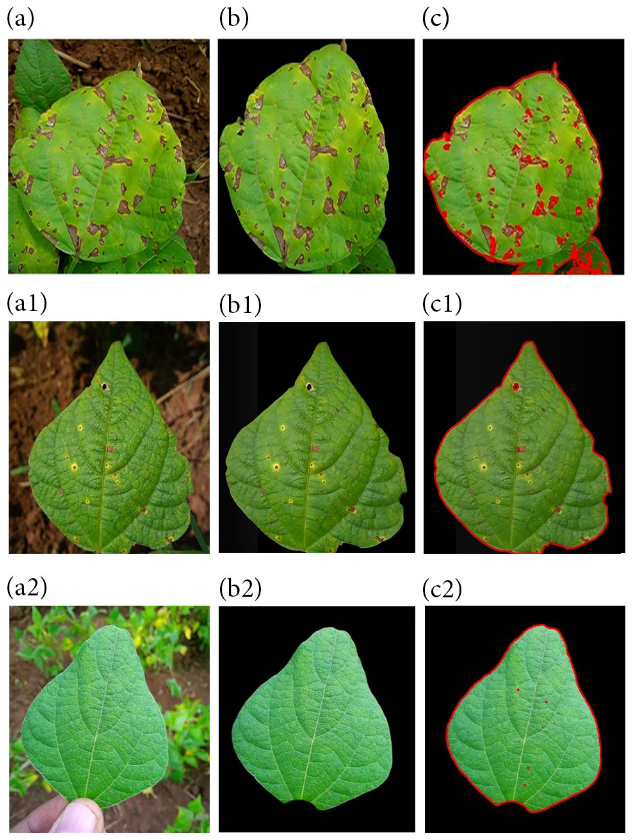


Figure 3
The preprocessing procedure used: (a) primary image, (b) background removal, and (c) binary threshold and contour detection



Finally, contour detection is used to extract object or region boundaries in an image. A contour refers to a curve connecting points along a consistent intensity edge, as shown in Equation (2) [19]. This analysis reveals the shape, size, and spatial distribution of the infected regions. Such geometric and morphological features can train classifiers or can be inputted to deep learning models for automated disease diagnosis. In this study, the method effectively highlighted high circularity and small contours on Br and irregular shapes on Als.

The contour detection method is summarized in the following algorithm.

Algorithm

Step 1.	Load Image:	Read the RGB image with no background from disk using OpenCV.
Step 2.	Convert to Grayscale:	Convert the RGB image to grayscale to simplify further processing
Step 3.	Apply Binary Thresholding:	• Convert the grayscale image to binary using a fixed threshold. • Pixels above the threshold are set to 255 (white), and those below to 0 (black).
Step 4.	Find Contours:	Detect contours from the binary image using tree-based contour retrieval and simple chain approximation.
Step 5.	Copy Original Image:	Make a copy of the original color image for drawing purposes.
Step 6.	Draw Contours:	Draw the detected contours on the copied image using blue color (255, 0, 0) and line thickness of 1.
Step 7.	Save the Image:	Save the image to destination folder.

2.3. Convolutional neural network

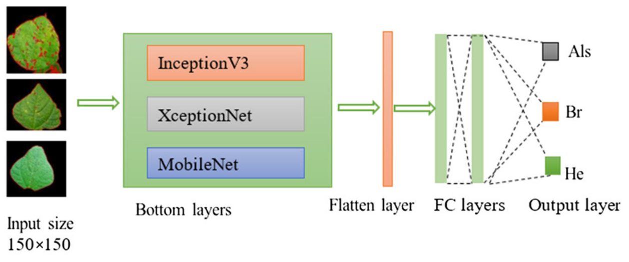
CNNs play a crucial role in deep learning and are widely used in CV tasks. They recognize visual patterns across image pixels with minimal preprocessing, making them highly effective in image-based detection studies. This section outlines the CNN models used in this research. GoogLeNet offers an alternative to CNNs by incorporating an inception module that introduces additional layers. GoogLeNet utilizes a replacement of average pooling for fully connected (FC) layers situated at the uppermost section of the convolutional network, resulting in a significant reduction in the quantity of parameters [20]. The architectural design of GoogLeNet was developed with the objective of minimizing energy consumption and memory utilization. This study also investigates InceptionV3, which is anticipated to succeed InceptionV1 and InceptionV2 [21].

Furthermore, it is worth noting that XceptionNet exhibited superior performance compared to InceptionV3, despite both models having an equal number of parameters. This notable difference in performance was observed specifically on the largest dataset. In this model, the conventional inception modules of GoogLeNet were substituted with depth-wise separable convolutions, which were subsequently followed by point-specific convolutions (1 × 1 convolutions) [22].

MobileNet is a paradigm that has been put up by Google as a solution for mobile and embedded systems, characterized by its lightweight nature. The utilization of depth-wise separable convolutions in the model leads to a reduction in the number of parameters necessary for network training. Additionally, it facilitates the construction of thin deep neural networks in comparison to the GoogLeNet and InceptionV3 models [23].

The CNN architecture consists of three main types of layers: convolutional, pooling, and FC layers. Feature extraction is performed using convolutional and pooling layers, and the FC layer maps these features to the final output, typically through a softmax function. As shown in Figure 4, this study employed three transfer learning CNN-based models, namely, InceptionV3, XceptionNet, and MobileNet, to train, validate, and test both the original and enhanced datasets. During training, the lower layers were frozen for feature extraction and the upper layers were fine-tuned for classification. Input images were resized to 150 × 150 pixels, and the FC layer contained 128 neurons, followed by a softmax layer with three output classes.

Figure 4
Proposed CNN architecture



3. Experimental Setup and Evaluation Parameters

Figure 5 shows the suggested architecture with a 500 × 500-pixel input image. The input images are sent via the background removal block to maintain the needed section before being converted to grayscale. Grayscale images aid the contour detection approach in detecting forms generated by bacteria or pests. The binary threshold block aids in distinguishing between image hues. The images are then subjected to contour detection to identify the various shapes on the images. This block's images are transformed to 150 × 150 pixels for faster processing and then trained using the CNN-based models InceptionV3, XceptionNet, and MobileNet to test and compare their performance. After testing the CNN-based models on the two sets of data, MobileNet trained on the enhanced data outperformed the others with an average accuracy of 96.69% and average F1-score of 0.97. Training was carried out using a 32 fixed batch size, a learning rate of 0.001, and a dropout rate of 0.05. Various augmentation approaches were employed to enhance the dataset and the overall learning process and performance. These techniques included random vertical and horizontal flipping, 90-degree rotation, and zooming [24–26]. Furthermore, the Adam optimizer was employed because Sembiring et al. [10] and Gui et al. [11]] demonstrated that it performed better in the dataset that we used in this work. All models were trained on a system outfitted with an Intel Core i5-7200u CPU (2.50 and 2.71 GHz) and 16 GB of memory.

We use four metrics to evaluate the classification models, as shown in Equations (3)–(6): accuracy (Ay), precision (Pn), recall (Rl), and F1-score (Fs).

$$A_y = \frac{T_{ne}+T_{ve}}{T_{ne}+T_{ve}+F_{ve}+F_{ne}} \tag{3}$$

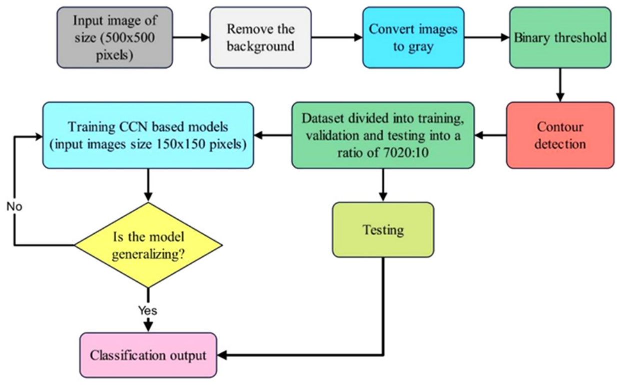
$$P_n = \frac{T_{ve}}{T_{ve}+F_{ve}} \tag{4}$$

$$R_l = \frac{T_{ve}}{T_{ve}+F_{ne}} \tag{5}$$

$$F_s = 2((P_n \cdot R_l)/(P_n + R_l)) \tag{6}$$

The term "true negative (Tne)" denotes the correct identification of the negative class by the model. Conversely, "true positive (Tve)" indicates a state in which the model correctly identifies the positive class. "False negative (Fne)" signifies an incorrect prediction made by the model regarding the negative class. Moreover, "false positive (Fve)" describes a case where the model mistakenly predicts a positive class.

Figure 5
Framework of the proposed method


Input image of size (500x500 pixels)
Remove the background
Convert images to gray
Binary threshold
Contour detection
Dataset divided into training, validation and testing into a ratio of 7020:10
Training CCN based models (input images size 150x150 pixels)
No
Is the model generalizing?
Yes
Testing
Classification output

4. Experimental Results and Discussion

This section describes the experimental findings, covering the training accuracy and losses for models trained on two separate datasets. It also evaluates each model's confusion matrix and compares the results of the proposed method to those of existing methodologies.

Figure 6 shows the average accuracy and loss of the models trained on both original and enhanced images. These averages were calculated by taking the mean of the training and validation accuracies, as well as the losses, throughout all epochs. In this study, the models trained on original images are referred to as InceptionV3, XceptionNet, and MobileNet, and those trained on improved images are referred to as InceptionV3-en, XceptionNet-en, and MobileNet-en. The orange line in the graph presents a model trained on enhanced images, whereas the blue line indicates a model trained on modified images. Figures 6(a) and (b) show the average accuracy and loss of the models trained on InceptionV3 and InceptionV3-en, respectively. The average accuracy rates of InceptionV3 and InceptionV3-en were 79.53% and 82.78%, respectively, with average losses of 0.45 and 0.36. Figures 6(c) and (d) illustrate the average accuracy and loss of the models trained on XceptionNet and XceptionNet-en, respectively. The average accuracy rates of XceptionNet and XceptionNet-en were 83.65% and 93.84%, respectively, with average losses of 0.35 and 0.14. Finally, Figures 6(e) and (f) show the average accuracy and loss of the models trained on MobileNet and MobileNet-en, respectively. The average accuracy rates of MobileNet and MobileNet-en were 93.72% and 96.69%, respectively, with average losses of 0.15 and 0.11.

Figure 7 compares the six models' average accuracies. The graph clearly demonstrates that MobileNet-en surpasses all other models with an average accuracy of 96.69%, and InceptionV3 has the lowest average accuracy with 79.36%. In particular, when compared to existing methods, the suggested approach improves average accuracy by 2.97%–8.16%.

The confusion matrix of the models trained on original and enhanced images is presented in Table 1. The InceptionV3 model exhibited the lowest performance with an accuracy of 79.36%, followed by the XceptionNet model at 84.68% and the MobileNet model at 93.72%. For the InceptionV3 model, the precision, recall, and F1-score for the Br class were all recorded at 0.72, and these metrics for the other classes ranged from 0.74 to 0.92. The XceptionNet model also showed low precision, recall, and F1-score of 0.8 for the Br class, with values for the other classes varying between 0.81 and 0.93. In contrast, the MobileNet model demonstrated precision, recall, and F1-scores ranging from 0.90 to 0.96, with the He class achieving the highest score of 0.96, indicating a balanced detection of true positives and true negatives. Notably, all three models exhibited low precision for the Br class, and the He class performed better than the other classes in terms of recall.

In comparison, the InceptionV3-en, XceptionNet-en, and MobileNet-en models showed enhancements in performance of 3.35%, 8.16%, and 2.97%, respectively. The precision, recall, and F1-scores for the InceptionV3-en model ranged from 0.74 to 0.94, those for the XceptionNet-en model ranged from 0.88 to 1, and those for the MobileNet-en model ranged from 0.94 to 1. The F1-scores for all three classes in the MobileNet-en model surpassed those of the other models, demonstrating a favorable balance between true positives and false negatives. Moreover, the He class consistently achieved the highest F1-score of 0.98 across all classes. The experimental findings indicate that the application of preprocessing techniques led to an improvement in model performance by as much as 8.16%.

Table 2 compares the accuracy performance of the proposed models in our work to similar approaches in the literature. The data presented in the

Figure 6
The average accuracy and loss of the models trained on both original and enhanced images. (a) InceptionV3 and InceptionV3-en average accuracy, (b) InceptionV3 and InceptionV3-en average loss, (c) XceptionNet and XceptionNet-en average accuracy, (d) XceptionNet and XceptionNet-en average loss, (e) MobileNet and MobileNet-en average accuracy, and (f) MobileNet and MobileNet-en average loss

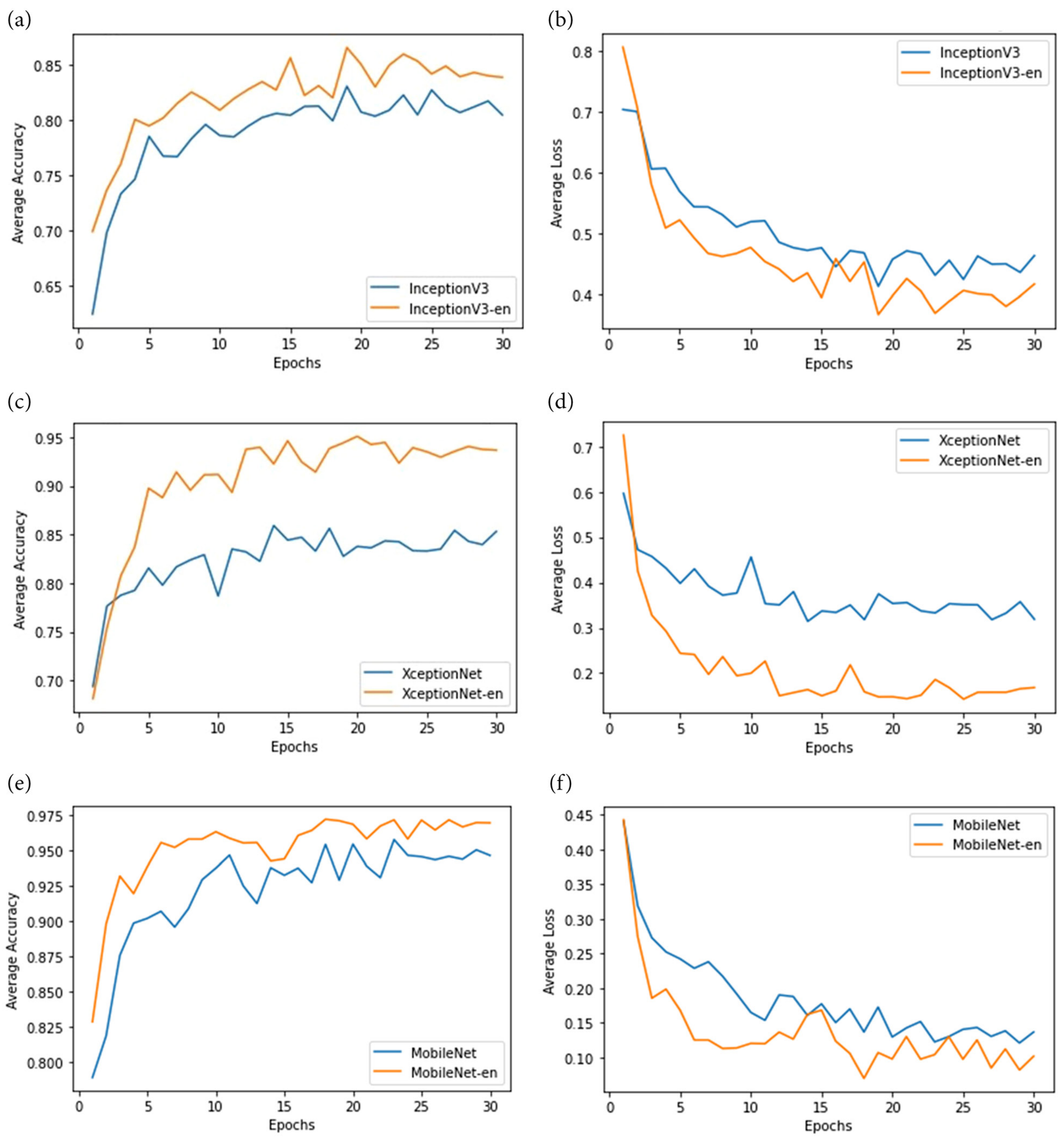


table demonstrate that the models proposed in this study exhibit superior performance compared to all previously suggested methodologies. The models that were suggested attained an accuracy of 96.69%.

5. Conclusion

This study employed a variety of preprocessing approaches and PR techniques to improve images collected in challenging conditions such as poor lighting, noise, or background complexity, ultimately improving the performance of ML models. PR methods were applied alongside CNN-based models to enhance the identification of disease-related characteristics on bean leaves. Two datasets were utilized for model training: the original bean images and the enhanced versions. The original images underwent preprocessing to eliminate the background. In contrast, the enhanced bean images experienced a three-step preprocessing process that involved background removal, binary filtering, and contour detection, effectively minimizing background noise. The average accuracy rates of the models trained

Figure 7
Comparison of accuracies of the models

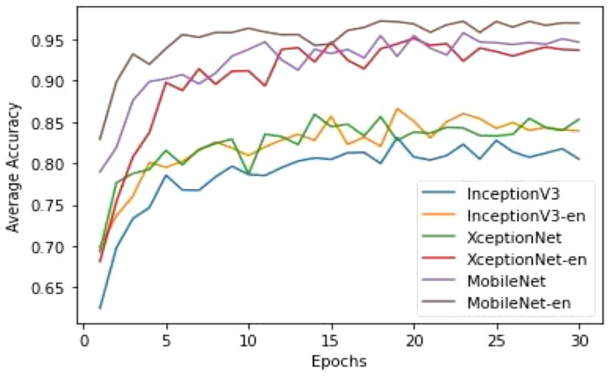


Table 1
Confusion matrix of all models

Class	Als	Br	He	A_y (%)	P_n	R_l	F_s	Improvement (%)
				InceptionV3				
Als	37	10	3	79.36	0.86	0.74	0.80	-
Br	6	36	8		0.72	0.72	0.72	
He	0	4	46		0.80	0.92	0.86	
				XceptionNet				
Als	41	8	1	84.68	0.80	0.82	0.81	-
Br	8	40	2		0.80	0.80	0.80	
He	2	2	46		0.94	0.92	0.93	
				MobileNet				
Als	46	3	1	93.72	0.96	0.92	0.94	-
Br	2	47	1		0.90	0.94	0.92	
He	0	2	48		0.96	0.96	0.96	
				InceptionV3-en				
Als	37	12	1	82.71	0.88	0.74	0.80	3.35
Br	4	40	6		0.74	0.80	0.77	
He	1	2	47		0.87	0.94	0.90	
				XceptionNet-en				
Als	44	2	4	92.84	0.96	0.88	0.92	**8.16**
Br	2	45	3		0.96	0.90	0.93	
He	0	0	50		0.88	1.00	0.94	
				MobileNet-en				
Als	47	2	1	**96.69**	**0.98**	**0.94**	**0.96**	2.97
Br	1	48	1		**0.96**	**0.96**	**0.96**	
He	0	0	50		**0.96**	**1.00**	**0.98**	

Table 2
Comparison of the proposed method to existing methods

Method used		Accuracy on test set (%)
Proposed	**MobileNet-en**	**96.69**
Existing	EfficientNetB6 [8]	91.74
	MobileNet [9]	92

on the enhanced images showed considerable improvement, reaching 82.71%, 92.84%, and 96.69%. In contrast, models that were trained on the original images achieved accuracy rates of 79.36%, 84.68%, and 93.72% for InceptionV3, XceptionNet, and MobileNet, respectively. Experimental results revealed that models utilizing enhanced images exhibited performance boosts of up to 8.16%. MobileNet-en performed particularly well, outperforming all other models with an average accuracy of 96.69% and an F1-score of 0.97.

A key limitation of this study is the sensitivity of the contour detection technique to background noise. Although the method is designed to identify irregular shapes in the foreground, it often misidentifies features when applied to images with complex or noisy backgrounds. As a result, its effectiveness is reduced in field conditions where background interference is common. Future studies will explore the impact of the contour recognition method on various crop types.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

The data that support the findings of this study are openly available in AI-Lab-Makerere at https://github.com/AI-Lab-Makerere/ibean/.

Author Contribution Statement

Farian S. Ishengoma: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration, Funding acquisition. **Joseph P. Telemala:** Software, Validation, Formal analysis, Writing – review & editing.

References

[1] Sarkar, C., Gupta, D., Gupta, U., & Hazarika, B. B. (2023). Leaf disease detection using machine learning and deep learning: Review and challenges. *Applied Soft Computing*, *145*, 110534. https://doi.org/10.1016/j.asoc.2023.110534

[2] Shoaib, M., Shah, B., Ei-Sappagh, S., Ali, A., Ullah, A., Alenezi, F., …, & Ali, F. (2023). An advanced deep learning models-based plant disease detection: A review of recent research. *Frontiers in Plant Science*, *14*, 1158933. https://doi.org/10.3389/fpls.2023.1158933

[3] Harakannanavar, S. S., Rudagi, J. M., Puranikmath, V. I., Siddiqua, A., & Pramodhini, R. (2022). Plant leaf disease detection using computer vision and machine learning algorithms. *Global Transitions Proceedings*, *3*(1), 305–310. https://doi.org/10.1016/j.gltp.2022.03.016

[4] Liu, J., & Wang, X. (2020). Tomato diseases and pests detection based on improved Yolo V3 convolutional neural network. *Frontiers in Plant Science*, *11*, 898. https://doi.org/10.3389/fpls.2020.00898

[5] Paithane, P., & Wagh, S. J. (2023). Novel modified kernel fuzzy c-means algorithm used for cotton leaf spot detection. *System Research and Information Technologies*, (4), 85–99. https://doi.org/10.20535/SRIT.2308-8893.2023.4.07

[6] Jung, M., Song, J. S., Shin, A. Y., Choi, B., Go, S., Kwon, S. Y., …, & Kim, Y. M. (2023). Construction of deep learning-based disease detection model in plants. Scientific Reports, *13*(1), 7331. https://doi.org/10.1038/s41598-023-34549-2

[7] Tian, H., Wang, T., Liu, Y., Qiao, X., & Li, Y. (2020). Computer vision technology in agricultural automation—A review. *Information Processing in Agriculture*, *7*(1), 1–19. https://doi.org/10.1016/j.inpa.2019.09.006

[8] Singh, V., Chug, A., & Singh, A. P. (2023). Classification of beans leaf diseases using fine tuned CNN model. *Procedia Computer Science*, *218*, 348–356. https://doi.org/10.1016/j.procs.2023.01.017

[9] Elfatimi, E., Eryigit, R., & Elfatimi, L. (2022). Beans leaf diseases classification using MobileNet models. *IEEE Access*, *10*, 9471–9482. https://doi.org/10.1109/ACCESS.2022.3142817

[10] Sembiring, A., Away, Y., Arnia, F., & Muharar, R. (2021). Development of concise convolutional neural network for tomato plant disease classification based on leaf images. *Journal of Physics: Conference Series*, *1845*(1), 012009. https://doi.org/10.1088/1742-6596/1845/1/012009

[11] Gui, J., Fei, J., Wu, Z., Fu, X., & Diakite, A. (2021). Grading method of soybean mosaic disease based on hyperspectral imaging technology. *Information Processing in Agriculture*, *8*(3), 380–385. https://doi.org/10.1016/j.inpa.2020.10.006

[12] Agarwal, M., Singh, A., Arjaria, S., Sinha, A., & Gupta, S. (2020). ToLeD: Tomato leaf disease detection using convolution neural network. *Procedia Computer Science*, *167*, 293–301. https://doi.org/10.1016/j.procs.2020.03.225

[13] Aravind, K. R., Raja, P., Aniirudh, R., Mukesh, K. V., Ashiwin, R., & Vikas, G. (2019). Grape crop disease classification using transfer learning approach. In *Proceedings of the International Conference on ISMAC in Computational Vision and Bio-Engineering 2018*, 1623–1633. https://doi.org/10.1007/978-3-030-00665-5_150

[14] Picon, A., Alvarez-Gila, A., Seitz, M., Ortiz-Barredo, A., Echazarra, J., & Johannes, A. (2019). Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild. *Computers and Electronics in Agriculture*, *161*, 280–290. https://doi.org/10.1016/j.compag.2018.04.002

[15] Yang, R., Wu, Z., Fang, W., Zhang, H., Wang, W., Fu, L., …, & Cui, Y. (2023). Detection of abnormal hydroponic lettuce leaves based on image processing and machine learning. *Information Processing in Agriculture*, *10*(1), 1–10. https://doi.org/10.1016/j.inpa.2021.11.001

[16] Pantazi, X. E., Moshou, D., & Tamouridou, A. A. (2019). Automated leaf disease detection in different crop species through image features analysis and One Class Classifiers. *Computers and Electronics in Agriculture*, *156*, 96–104. https://doi.org/10.1016/j.compag.2018.11.005

[17] Emwebaze, E. (2020). *AI-Lab-Makerere* [Data set]. GitHub. https://github.com/AI-Lab-Makerere/ibean

[18] Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing (4th ed.).* Malaysia: Pearson.

[19] Masoud, K. M., Persello, C., & Tolpekin, V. A. (2019). Delineation of agricultural field boundaries from Sentinel-2 images using a novel super-resolution contour detector based on fully convolutional networks. *Remote Sensing*, *12*(1), 59. https://doi.org/10.3390/rs12010059

[20] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826. https://doi.org/10.1109/CVPR.2016.308

[21] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778. https://doi.org/10.1109/CVPR.2016.90

[22] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1251–1258. https://doi.org/10.1109/CVPR.2017.195

[23] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

[24] Ishengoma, F. S., Rai, I. A., & Ngoga, S. R. (2022). Hybrid convolution neural network model for a quicker detection of infested maize plants with fall armyworms using UAV-based images. *Ecological Informatics*, *67*, 101502. https://doi.org/10.1016/j.ecoinf.2021.101502

[25] Taghavi Namin, S., Esmaeilzadeh, M., Najafi, M., Brown, T. B., & Borevitz, J. O. (2018). Deep phenotyping: Deep learning for temporal phenotype/genotype classification. *Plant Methods*, *14*(1), 66. https://doi.org/10.1186/s13007-018-0333-4

[26] Bakkouri, I., Afdel, K., Benois-Pineau, J., & Catheline, G. (2022). BG-3DM2F: bidirectional gated 3D multi-scale feature fusion for Alzheimer's disease diagnosis. *Multimedia Tools and Applications*, *81*(8), 10743–10776. https://doi.org/10.1007/s11042-022-12242-2

How to Cite: Ishengoma, F. S., & Telemala, J. P. (2025). Integrating Pattern Recognition and CNN-Based Models for Improved Bean Disease Detection and Agricultural Yield Enhancement. *Artificial Intelligence and Applications*, *3*(4), 385–391. https://doi.org/10.47852/bonviewAIA52024376