TSSA Journal of Tomography System & Sensors Application Vol. 8, Issue 2, 2025
www.tssa.com.my e-ISSN: 636-9133

Grape Leaf Disease Detection Using Convolutional Neural Network

Muhammad Faza Iqmal Mustafa Kamal[1], Aminurrashid Noordin[1*], Madiha Zahari[2], Ruzairi Abdul Rahim[3], Mohd Ariffanan Mohd Basri[3], Izzuddin Mat Lazim[4]

[1]Faculty of Electrical Technology and Engineering, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka.
[2]Faculty of Electronics & Computer Technology & Engineering, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka.
[3]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia
[4]Faculty of Engineering & Built Environment, Universiti Sains Islam Malaysia, Bandar Baru Nilai, 71800, Nilai, Negeri Sembilan, Malaysia

Corresponding author* email: aminurrashid@utem.edu.my
Available online 30 December 2025

ABSTRACT

This project focuses on developing a plant disease detection system using Convolutional Neural Networks (CNN) to address the critical challenge of identifying plant diseases early in agriculture. The proposed system leverages image analysis to classify diseases such as Grape Black Rot, Leaf Blight, and healthy conditions in grape leaves. Utilizing AlexNet architecture in MATLAB, the model processes a dataset of 500 leaf images (70% for training, 30% for testing) with image preprocessing techniques like resizing and normalization. The methodology involves designing a MATLAB-based GUI for user interaction, allowing image uploads, disease detection, affected area analysis, and remedy suggestions. Model performance was evaluated on multiple metrics, achieving an overall accuracy of 99.3% on the validation dataset. Tests on 60 samples consistently demonstrated high prediction confidence (96.42%-100%) and accurate classification of healthy and diseased leaves. Quantitative analysis of the affected area using clustering revealed detailed insights into disease severity, supporting effective decision-making. This system shows strong potential for real-time agricultural applications, contributing to sustainable farming practices and enhancing food security. Future enhancements include integrating mobile platforms for broader accessibility.

Keywords: CNN, AlexNet, Grape Leaf, Black Rot, Leaf Blight

1. Introduction

Artificial Neural Networks (ANNs) are mathematical models that simulate the structure and functioning of biological neural systems, processing data in a distributed and parallel manner. By adjusting the synaptic weights between interconnected neurons, ANNs learn to perform tasks such as pattern recognition and classification. A specialized form of ANN, known as a Convolutional Neural Network (CNN), consists of multiple layers, including convolutional and pooling layers, and has demonstrated remarkable performance across various domains, particularly in image processing applications [1].

The foundational concept of CNNs is rooted in the biological mechanisms of the mammalian visual cortex. In the 1960s, Hubel and Wiesel conducted pioneering studies on the visual processing system of cats, revealing that individual neurons respond to specific spatial regions in the visual field termed receptive fields [2]. Their contributions, which earned the Nobel Prize in Physiology or Medicine in 1981, laid the groundwork for artificial vision systems [3]. Building upon this, Fukushima introduced the Neocognitron in 1980, a hierarchical neural network architecture designed for pattern recognition, incorporating concepts such as local receptive fields and weight sharing—principles that are integral to modern CNN architectures [4].

Subsequent advancements in CNNs were marked by the development of LeNet-5 by LeCun et al. in the late 1980s and early 1990s. LeNet-5, designed for handwritten digit recognition, was among the first CNN models successfully applied to the MNIST dataset. It featured key architectural components now common in deep learning: convolutional layers, subsampling (pooling) layers, and fully connected layers.

A breakthrough occurred in 2012 with the introduction of AlexNet by Krizhevsky, Sutskever, and Hinton. This deep CNN architecture significantly outperformed its competitors in the ImageNet Large Scale Visual Recognition Challenge

(ILSVRC), heralding a new era in deep learning. AlexNet popularized the use of ReLU (Rectified Linear Unit) activations, dropout for regularization, overlapping max pooling, and GPU acceleration, which collectively enabled the training of large-scale models for image classification tasks.

Within the domain of object detection, CNN-based methods are commonly categorized into two-stage and one-stage approaches. The two-stage approach involves a sequential process of generating region proposals followed by classification. Representative models of this category include R-CNN, Fast R-CNN, and Faster R-CNN [5]. In contrast, one-stage methods such as You Only Look Once (YOLO) [6] and Single Shot MultiBox Detector (SSD) [7] streamline the detection process by performing localization and classification in a single step. Although generally faster and more computationally efficient, one-stage models may sacrifice some degree of accuracy compared to two-stage counterparts.

Recent works have further demonstrated the effectiveness of CNNs in specialized image-based tasks. For example, Amin et al. [8] proposed a robust model for the automated detection and classification of corn leaf diseases. Leveraging feature fusion between two pre-trained networks EfficientNetB0 and DenseNet121. Their model classifies images of corn leaves into categories including gray leaf spot, common rust, northern leaf blight, and healthy foliage. This end-of-end system addresses limitations associated with conventional large parameter models and illustrates the transformative potential of deep learning in agricultural disease management. Their approach achieved a classification accuracy of 98.56%, outperforming other CNN-based methods and reinforcing the viability of CNNs for high-precision plant disease diagnostics.

Moupojou et al. introduced FieldPlant, a dataset with 5,170 real-field images and 8,629 annotated leaves across 27 disease classes, addressing limitations in existing datasets. Using the RoboFlow platform, the dataset enables real-time plant disease detection via deep learning. Their hybrid model achieved 95% accuracy on tomato disease classification. Khattak et al. reported 94.55% accuracy using a 2-layer CNN for citrus disease detection. Additionally, Moupojou et al. applied multi-task learning, achieving 84.71% plant and 75.06% disease classification accuracy on the PlantDoc dataset [9].

Similarly, Restrepo-Arias et al. [10] explored plant disease detection through image texture analysis combined with Bayesian optimization and lightweight CNN architectures. By preprocessing and segmenting plant images to mitigate morphological bias, the study successfully trained compact CNN models such as MobileNet, SqueezeNet, and NasNetMobile. These models attained competitive accuracy levels ranging from 91.50% (ShuffleNet) to 96.31% (MobileNet) while maintaining low computational complexity, making them suitable for resource-constrained environments.

Building upon these prior developments, this study applies the AlexNet architecture for the detection of Black Rot and Leaf Blight in grapevine leaves using MATLAB. AlexNet, consisting of eight layers (five convolutional and three fully connected), introduced several architectural innovations that remain central to CNN design. These include ReLU activations, dropout, overlapping max pooling, and GPU-enabled training. The model processes 227×227 RGB images, making it well-suited for high-resolution image analysis. Its proven performance in large-scale image classification tasks underscores its potential for reliable plant disease detection, thereby contributing to precision agriculture and crop health monitoring.

2. Methodology

The methodology for this project adopts a structured, phased approach to ensure systematic development, validation, and deployment of the CNN-based plant disease classification system. Each phase builds on the outcomes of the previous, ensuring thoroughness and traceability throughout the development lifecycle. Publicly available plant disease datasets, such as PlantVillage, serve as a primary source of training data. These datasets, often found on platforms like Kaggle or academic repositories, contain high-resolution images of plant leaves annotated with disease types and health status. However, they may suffer from certain limitations, such as class imbalance and artificial lighting conditions, which may not fully represent field environments.

2.1 Disease Detection Process Using CNN

The detailed pipeline for CNN-based disease detection begins with the acquisition of a labeled dataset of plant leaf images, categorized into healthy and diseased classes. These images undergo pre-processing, which includes resizing, normalization, and format conversion to standardize input dimensions and enhance compatibility with the CNN model. The cleaned dataset is then divided into training and testing subsets.

The CNN model is trained using pre-processed training images to learn discriminative features corresponding to each category. During inference, a test image unseen during training is subjected to the same pre-processing steps and is then passed through the trained CNN for classification. If the model detects a disease, it identifies the specific condition and displays corresponding remedies or recommendations. If no disease is detected, the image is classified as healthy. This process concludes with the output of the classification result.

2.2 CNN Training Pipeline Using AlexNet in MATLAB

Figure 1 shows algorithm training to set up the AlexNet training pipeline to initialize the necessary variables. This includes specifying the path to the dataset folder using a variable called dataFolder. Additionally, the input size for the images is defined to match the requirements of AlexNet, which expects images of size 227 x 227 pixels. These constants and paths ensure that the model and data preprocessing align correctly, forming the foundation for subsequent operations. To prepare the data for training, the dataset is loaded into MATLAB using the imageDatastore function. This automatically assigns labels to images based on the folder names. The dataset is then divided into training and validation sets, with 70% allocated for training (imdsTrain) and 30% for validation (imdsValidation). Data augmentation is applied to the training set to introduce variability, such as random reflections and scaling, improving the model's robustness. Meanwhile, the validation set is resized to the required input dimensions without augmentation.

Before training, it's helpful to visualize the training data to confirm that the data loading process was successful. A random selection of nine images from the training dataset is displayed in a grid. This optional but informative step ensures the images are correctly labeled and formatted, giving confidence in the correctness of the data pipeline before proceeding with model training.

AlexNet, a pre-trained deep learning model, is customized to fit the specific dataset. The base layers of AlexNet are retained, excluding the final three layers, which are tailored to its original dataset. These layers are replaced with a fully connected layer with neurons matching the number of dataset classes, a softmax layer for probability outputs, and a classification layer for label prediction. This modification enables AlexNet to specialize in the new dataset while leveraging its pre-trained features.

Configuring training parameters is crucial for optimizing performance. The Stochastic Gradient Descent with Momentum (SGDM) optimizer is selected for efficient training. Key parameters include a mini-batch size of 32 images, a maximum of 10 epochs to limit the number of passes through the dataset, and an initial learning rate of 0.0001 to ensure steady convergence. Validation data is incorporated into the training process to monitor the model's performance and prevent overfitting.

Using the configured parameters and the modified AlexNet structure, the model is trained with the augmented training data. This step involves updating the model's weights to minimize the loss function, thereby improving its ability to classify images accurately. After training, the resulting model, named netTransfer, is saved as a.mat file (PDC_Train.mat) for future use, preserving the trained network and its parameters.

The model's performance is evaluated on the validation set to measure its accuracy. Validation images are classified using the trained model, and the predictions are compared against the actual labels. The resulting accuracy percentage quantifies the model's ability to generalize to unseen data. This step provides insights into the model's reliability and highlights areas where it may require further fine-tuning.

Finally, the model's predictions are visualized to assess its real-world application. Four random images from the validation set are selected and displayed along with their predicted class labels. This step demonstrates the model's ability to correctly classify images, offering a qualitative view of their effectiveness and helping identify specific cases where it may struggle. Figure 2 displays the predictions of a machine learning model classifying grapevine leaf health. It includes four samples: the top two leaves are predicted to be healthy, showing no visible disease symptoms, while the bottom-left leaf is identified as having "Grape Black Rot," characterized by dark lesions, and the bottom-right leaf is predicted to have "blight," marked by browning and damage.

2.3 Model Evaluation and Visualization

Post-training, the model is evaluated using the validation dataset to assess its generalization capability. Each validation image is classified, and the predicted labels are compared with the ground truth to compute the classification accuracy. This quantitative metric reflects the model's effectiveness in real-world scenarios and identifies areas that may require further optimization.

To complement the quantitative results, qualitative evaluation is performed by visualizing model predictions. As shown in Figure 2, four random images from the validation set are displayed alongside their predicted class labels. The top two images are correctly classified as healthy, while the bottom-left is identified as having Grape Black Rot, evidenced by dark lesions, and the bottom-right is diagnosed with blight, characterized by browning and tissue degradation. This visualization provides intuitive validation of the model's predictive power and helps reveal potential misclassification patterns.

Algorithm 1: Training Process Using AlexNet

Input: Dataset directory (dataFolder)
Output: Trained AlexNet model (PDC_Train.mat)

1. **Initialize Variables**
   - Set dataFolder to the path of the dataset folder.
   - Define inputSize = [227, 227].
2. **Load and Preprocess Data**
   - Load dataset:
     imds ← Load images from dataFolder with labels based on folder names.
   - Split data into training and validation sets:
     [imdsTrain, imdsValidation] ← Split imds into 70% training and 30% validation.
   - Create image augmenter:
     imageAugmenter ← Configure augmenter with random reflection and scaling.
   - Apply augmentation:
     - augmentedTrain ← Resize and augment images from imdsTrain.
     - augmentedValidation ← Resize images from imdsValidation.
3. **Visualize Training Data**
   - For i from 1 to 9:
     - Randomly select an image from imdsTrain.
     - Display the image.
4. **Modify Pre-trained AlexNet**
   - Load pre-trained model:
     net ← Load AlexNet.
   - Extract base layers:
     layersTransfer ← All layers except last three of net.
   - Get number of classes:
     numClasses ← Count unique labels in imdsTrain.
   - Add new layers:
     - Fully connected layer with numClasses neurons.
     - Softmax layer.
     - Classification layer.
5. **Set Training Options**
   - Define training options:
     - Use stochastic gradient descent (SGDM).
     - Set MiniBatchSize = 32.
     - Set MaxEpochs = 10.
     - Set InitialLearnRate = 1e-4.
     - Include validation data.
6. **Train the Model**
   - Train network:
     netTransfer ← Train with augmentedTrain, layers, and training options.
   - Save trained model:
     Save netTransfer as PDC_Train.mat.
7. **Evaluate the Model**
   - Classify validation images:
     [YPred, scores] ← Classify augmentedValidation using netTransfer.
   - Compute accuracy:
     accuracy ← Mean of correct predictions.
   - Display accuracy.
8. **Display Predictions**
   - Randomly select 4 images from imdsValidation.
   - For each image:
     - Display the image.
     - Display the predicted label.

Figure 1. Algorithm of training process using AlexNet

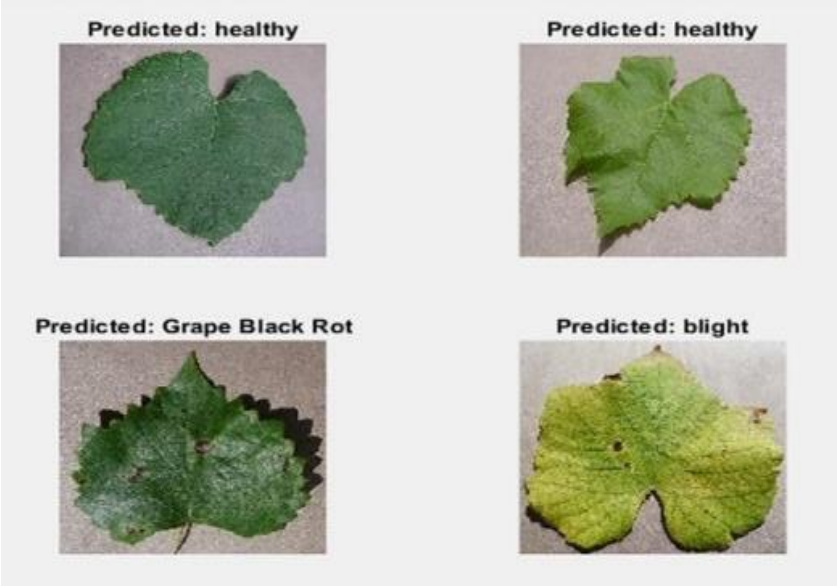

Figure 2. Sample Prediction using AlexNet Algorithm

3. Results and Discussion

This section presents the outcomes of our study on plant disease detection using CNNs implemented in MATLAB. Through detailed data analysis, we evaluate the CNN's performance across different types of diseases and compare its accuracy with other traditional methods. We also explore the impact of various preprocessing techniques and model hyperparameters on the results. This analysis helps identify the strengths and limitations of our approach.

Figures 3, 4 and 5 show the test dataset for the testing part. Each figure shows 20 datasets that had rename T1 to T60. Table 1 summarizes the results of a plant disease detection experiment across 20 test samples. Samples T1 to T6 are healthy, with no affected areas and 100% detection confidence. Samples T7 to T20 show varying percentages of disease spread across three clusters (indicating different disease types or severities). The detection was successful for all samples except T20, which had a lower confidence of 96.42%.

Overall, the detection algorithm performed well, consistently achieving high confidence levels (mostly 99.85-100%). Table 2 presents the results of plant disease detection for test samples T21 to T40. Samples T21 to T31 are healthy, with no affected areas (all percentages are 0.00%) and detection confidence of 100%. Samples T32 to T40 show varying degrees of disease spread across three clusters, representing different disease types or severities (e.g., "Black Rot" or "Leaf Blight"). The detection was successful for all samples, with confidence levels ranging from 96.42% to 100%. The results demonstrate consistent and accurate detection for both healthy and diseased samples, with high confidence in the predictions.

Table 3 shows plant disease detection results for samples T41 to T60. Samples T41 and T42 are healthy, with no affected areas (all percentages are 0.00%) and 100% detection confidence. Samples T43 to T60 show varying degrees of disease spread across three clusters (indicating different disease types or severities like "Black Rot" or "Leaf Blight"). The detection was successful for all samples, with confidence levels ranging from 96.42% to 100%. The percentages highlight the distribution of the affected area among clusters, such as T43 with 55.82% in Cluster 3 and T44 with 60.53% in Cluster 1. The results demonstrate accurate detection for all samples, maintaining high prediction confidence.

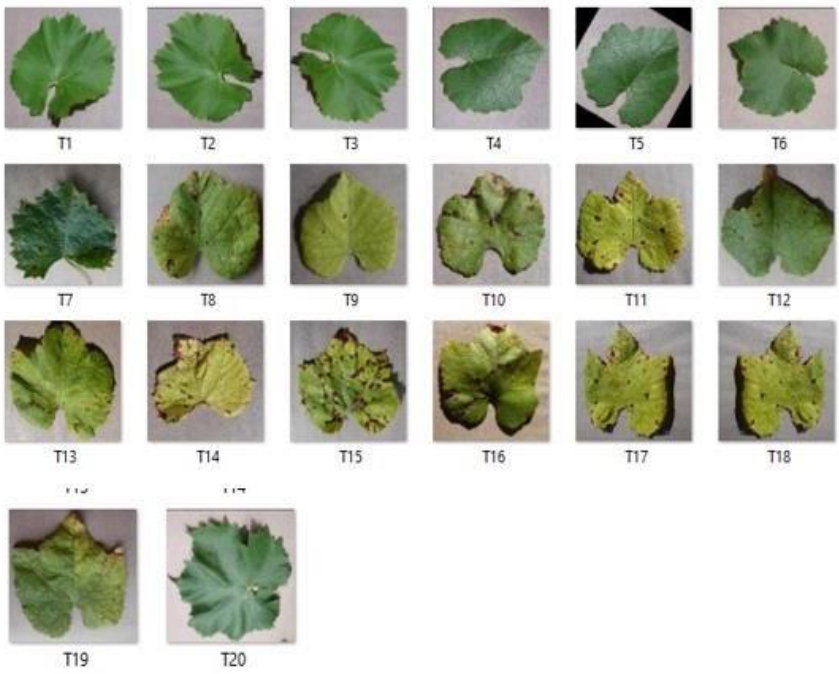

Figure 3. Test Dataset Part 1

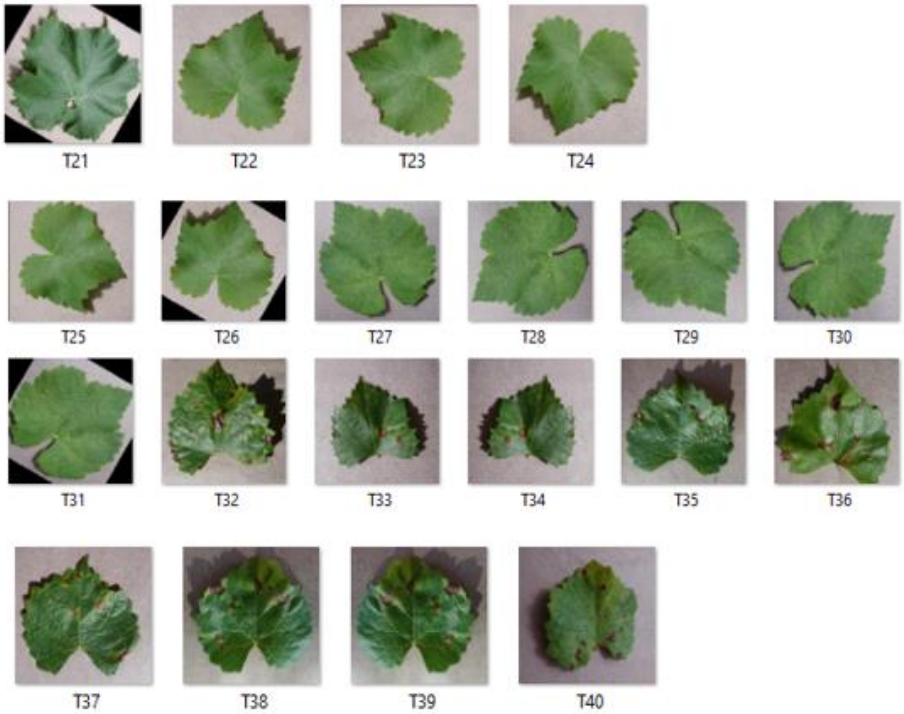

Figure 4. Test Dataset Part 2

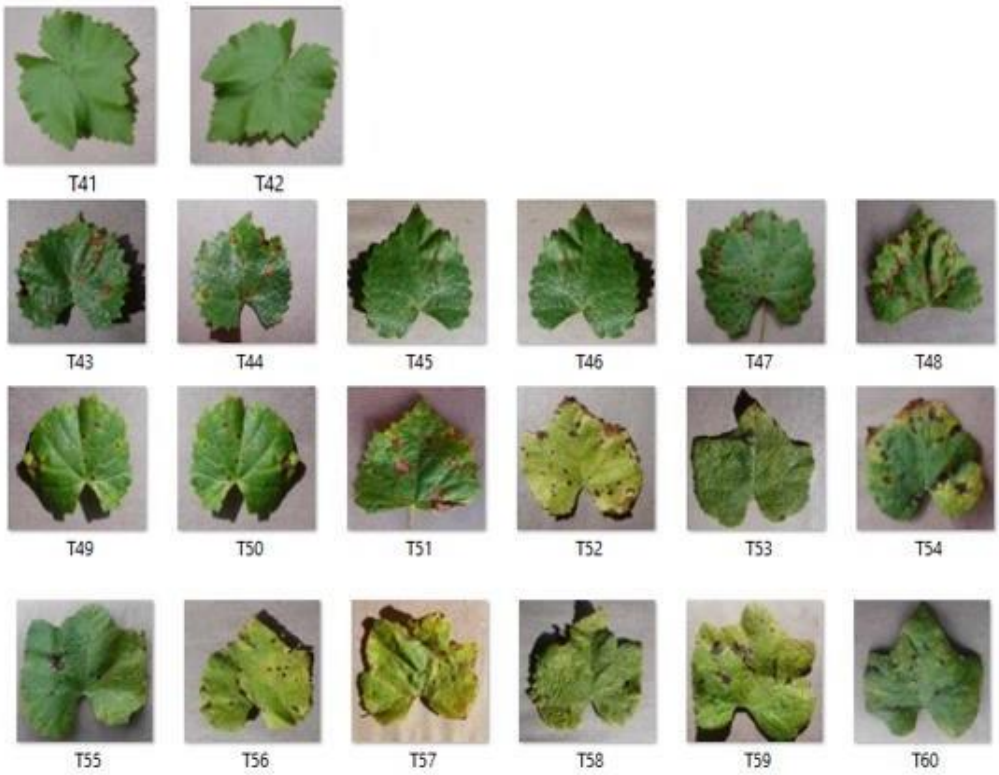

Figure 5. Test Dataset Part 3

The accuracy of the plant disease detection system was calculated for different categories, demonstrating its effectiveness in identifying healthy and diseased plants. For the healthy plant category, the system correctly identified 19 out of 20 images, resulting in an accuracy of $\frac{(20-1)}{20} x100\% = 95\%$. For the grape black rot and blight leaf categories, the system achieved perfect accuracy, correctly identifying all 20 images in each category, yielding 100% accuracy for both. When considering the overall performance across all categories, the system correctly classified 59 out of 60 images, leading to an overall accuracy of $\frac{(60-1)}{60} x100\% = 98.33\%$. These results highlight the robustness and reliability of the detection model in accurately distinguishing between healthy plants and various diseases.

Table 4 summarizes the dataset composition and the predicted confidence levels for a plant disease detection model. The training dataset consists of 200 healthy samples, 150 samples with grape black rot, and 150 samples with leaf blight. The validation/test dataset includes 20 samples for each category. The model demonstrates high predicted confidence in its classification performance, achieving 100% confidence for healthy samples and 99.8% confidence for both grape black rot and leaf blight. This indicates the model is highly reliable in detecting and classifying these conditions.

Table 1. Result of Plant Disease Detection Part 1

Test Data	Classify			Percentage of Affected Area			Detection Successful	Predicted Confidence (%)
	Healthy	Black Rot	Leaf Blight	Cluster 1	Cluster 2	Cluster 3		
T1				0.00	0.00	0.00	Yes	100.00
T2				0.00	0.00	0.00	Yes	100.00
T3				0.00	0.00	0.00	Yes	100.00
T4				0.00	0.00	0.00	Yes	100.00
T5				0.00	0.00	0.00	Yes	100.00
T6				0.00	0.00	0.00	Yes	100.00
T7				18.29	49.22	32.42	Yes	100.00
T8				48.55	36.73	14.72	Yes	100.00
T9				47.79	46.69	5.52	Yes	100.00
T10				37.59	46.09	16.33	Yes	100.00
T11				49.15	39.60	11.25	Yes	100.00
T12				32.95	22.03	45.03	Yes	99.94
T13				41.04	41.86	17.10	Yes	100.00
T14				56.26	11.11	32.63	Yes	99.97
T15				36.06	51.56	12.38	Yes	99.99
T16				22.17	51.16	26.67	Yes	99.85
T17				35.44	55.05	9.52	Yes	100.00
T18				55.17	34.95	9.89	Yes	100.00
T19				44.89	24.30	30.81	Yes	100.00
T20				36.91	40.43	22.65	No	96.42

Table 2. Result of Plant Disease Detection Part 2

Test Data	Classify			Percentage of Affected Area			Detection Successful	Predicted Confidence (%)
	Healthy	Black Rot	Leaf Blight	Cluster 1	Cluster 2	Cluster 3		
T21				0.00	0.00	0.00	Yes	100.00
T22				0.00	0.00	0.00	Yes	100.00
T23				0.00	0.00	0.00	Yes	100.00
T24				0.00	0.00	0.00	Yes	100.00
T25				0.00	0.00	0.00	Yes	100.00
T26				0.00	0.00	0.00	Yes	100.00
T27				0.00	0.00	0.00	Yes	100.00
T28				0.00	0.00	0.00	Yes	100.00
T29				0.00	0.00	0.00	Yes	100.00
T30				0.00	0.00	0.00	Yes	100.00
T31				0.00	0.00	0.00	Yes	100.00
T32				24.87	52.74	22.38	Yes	99.94
T33				66.61	16.86	16.53	Yes	100.00
T34				16.29	66.62	16.59	Yes	99.97
T35				53.46	19.74	26.27	Yes	99.99
T36				29.25	50.07	20.64	Yes	99.85
T37				26.60	51.86	21.54	Yes	100.00
T38				49.20	32.33	18.41	Yes	100.00
T39				18.48	49.21	32.26	Yes	100.00
T40				62.38	15.86	21.76	Yes	96.42

Table 3. Result of Plant Disease Detection Part 3

Test Data	Classify			Percentage of Affected Area			Detection Successful	Predicted Confidence (%)
	Healthy	Black Rot	Leaf Blight	Cluster 1	Cluster 2	Cluster 3		
T41				0.00	0.00	0.00	Yes	100.00
T42				0.00	0.00	0.00	Yes	100.00
T43				17.84	26.31	55.82	Yes	100.00
T44				60.53	16.46	23.01	Yes	100.00
T45				33.11	50.34	16.52	Yes	100.00
T46				16.54	50.36	33.07	Yes	100.00
T47				42.16	47.22	10.61	Yes	100.00
T48				61.39	8.77	29.84	Yes	100.00
T49				32.98	53.40	13.61	Yes	100.00
T50				13.71	53.42	32.87	Yes	100.00
T51				11.74	35.99	52.27	Yes	100.00
T52				51.55	9.35	59.10	Yes	99.94
T53				18.15	49.72	32.12	Yes	100.00
T54				17.21	30.54	52.85	Yes	99.97
T55				27.10	35.67	37.19	Yes	99.99
T56				53.06	8.66	38.28	Yes	99.85
T57				15.81	33.62	50.57	Yes	100.00
T58				51.96	30.57	17.46	Yes	100.00
T59				18.98	38.86	42.15	Yes	100.00
T60				48.77	26.52	24.70	Yes	96.42

Table 4. Predicted Confidence Comparison

	Healthy	Grape Black Rot	Leaf Blight
Train Dataset	200	150	150
Valid / Test Dataset	20	20	20
Predicted Confidence (%)	100.00	99.80	99.80

4. Conclusions

In conclusion, the project on plant disease detection using CNNs within the MATLAB framework has made significant progress in achieving its goals. By leveraging deep learning techniques and robust image processing capabilities, the project has developed a solution for identifying and classifying plant diseases with high accuracy. The structured approach to project planning, as outlined in the methodology section, has ensured thorough and efficient execution of each phase, from initialization to deployment. The use of CNNs alongside MATLAB's versatility has resulted in a sophisticated tool addressing critical agricultural challenges. Future works demonstrate a commitment to enhancing functionality, reflecting a forward-looking approach to innovation. Overall, the project utilizes cutting-edge technology to revolutionize crop management practices, offering timely and precise disease detection that holds immense promise for agricultural productivity. The findings highlight the model's outstanding accuracy in identifying and classifying plant diseases, showcasing its effectiveness compared to traditional methods and its potential to assist farmers in early disease management.

Acknowledgment

The authors would like to thank, Fakulti Teknologi dan Kejuruteraan Elektrik (FTKE), Universiti Teknikal Malaysia Melaka (UTeM), Ministry of Higher Education and the Advanced Academia-Industry Collaboration Laboratory (AiCL) UTeM for supporting this research.

References

[1] X. Chen, "The study for convolutional neural network and corresponding applications," Theoretical and Natural Science, vol. 5, no. 1, pp. 182–187, May 2023, doi: 10.54254/2753-8818/5/20230387.

[2] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," The Journal of Physiology, vol. 160, no. 1, pp. 106–154, Jan. 1962, doi: 10.1113/jphysiol.1962.sp006837.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[4] K. Fukushima, "Artificial Vision by Deep CNN Neocognitron," IEEE Transactions on Systems Man and Cybernetics Systems, vol. 51, no. 1, pp. 76–90, Jan. 2021, doi: 10.1109/tsmc.2020.3042785.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, Jun. 2016, doi: 10.1109/tpami.2016.2577031.

[6] C.-Y. Cao, J.-C. Zheng, Y.-Q. Huang, J. Liu, and C.-F. Yang, "Investigation of a promoted you only look once algorithm and its application in traffic flow monitoring," Applied Sciences, vol. 9, no. 17, p. 3619, Sep. 2019, doi: 10.3390/app9173619.

[7] W. Liu et al., "SSD: Single Shot MultiBox Detector," in Lecture notes in computer science, 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.

[8] H. Amin, A. Darwish, A. E. Hassanien, and M. Soliman, "End-to-End Deep learning model for corn leaf disease classification," IEEE Access, vol. 10, pp. 31103–31115, Jan. 2022, doi: 10.1109/access.2022.3159678.

[9] E. Moupojou et al., "FieldPlant: A dataset of field plant images for plant disease detection and classification with deep learning," IEEE Access, vol. 11, pp. 35398–35410, Jan. 2023, doi: 10.1109/access.2023.3263042.

[10] J. F. Restrepo-Arias, J. W. Branch-Bedoya, and G. Awad, "Plant Disease Detection Strategy Based on Image Texture and Bayesian Optimization with Small Neural Networks," Agriculture, vol. 12, no. 11, p. 1964, Nov. 2022, doi: 10.3390/agriculture12111964.