Journal of Advanced College of Engineering and Management

Visual Interpretation and Classification of Apple Leaf Diseases via Grad-CAM and Convolutional Neural Networks

[1*]Mingma Gyaltsen Lama Sherpa, [2]Rajad Shakya
[1,2]Department of Computer and Electronics Engineering, Institute of Engineering, Thapathali Campus, Kathmandu, Nepal
Email: rshakya.8063@tcioe.edu.np
Corresponding email: [*]mingmargylsten@gmail.com
DOI: 10.3126/jacem.v12i01.93931

Abstract
Apple cultivation is a crucial agricultural activity in various mountainous regions, playing a vital role in supporting the local economy and sustaining the livelihoods of farmers. Several prominent mountain districts are known for leading apple production. However, apple orchards in these areas are often threatened by numerous diseases that reduce fruit yield and quality. In this research, we suggest a machine learning-based technique to automate the detection and classification of common apple diseases based on images of apple leaves collected from various regions. Through the use of Convolutional Neural Networks (CNN), the system can classify diseases with 97.36% precision. For post hoc explainability, Grad-CAM is used, which highlights the important regions that influenced CNN's decision. The automated disease detection tool provides farmers in Nepal's rural mountain areas with an affordable real time solution to monitor orchard health, minimize crop loss, and improve apple production. The dataset used in this study is originally derived from the United States based PlantVillage dataset, which is widely used for apple leaf disease classification research. Although the dataset is not collected from Nepal, the visual characteristics of apple leaf diseases remain largely consistent across regions due to similar biological infection patterns. Therefore, the model trained on this dataset is applicable to Nepali apple cultivation environments as well. At present, a publicly available or annotated Nepali specific apple leaf disease dataset is not available, which limits region-specific training and evaluation.

Keywords*—Apple disease detection, Agriculture, CNN, Deep Learning, Grad-CAM, Machine Learning*

1. INTRODUCTION

Apple fruit is highly susceptible to numerous diseases that adversely affect the global agricultural sector and overall production. Despite the potential for high yields, the average national production of apples remains relatively low [7]. Apple cultivation plays a vital role in the agricultural economy of many countries, including Nepal. However, the productivity and quality of apple crops are significantly impacted by various diseases, such

as Apple Scab, Black Rot, and Cedar Apple Rust. Traditional methods of disease identification are heavily relying on manual inspection, which is time-consuming, labor-intensive, and prone to human error. As the scale of apple production expands, there is a growing need for accurate and automated disease detection systems. A brief description of these diseases is provided below.

Black rot is a fungal infection caused by Diplodia seriata fungus. It develops small sneaks on the upper surface of the leaf during leaf unfolding. Later, frogeye spots with reddish or purplish edges appear on the infected leaves. As the lesion ages, it becomes chlorotic and higher severity leads to defoliation weakening the tree. Black rot affects both fruits and leaves of the tree.

Scab is a serious infection transmitted through Venturia inaequalis fungus. Disease symptoms appear as pale yellow or olive green spots on the upper surface and as velvety or dark lesions on the lower surface of the leaf. As the infection spreads, it causes leaves to curl up or drop. The severe infection leads to continuous defoliation and damage to the tree. Scab also affects both fruits and leaves of the tree.

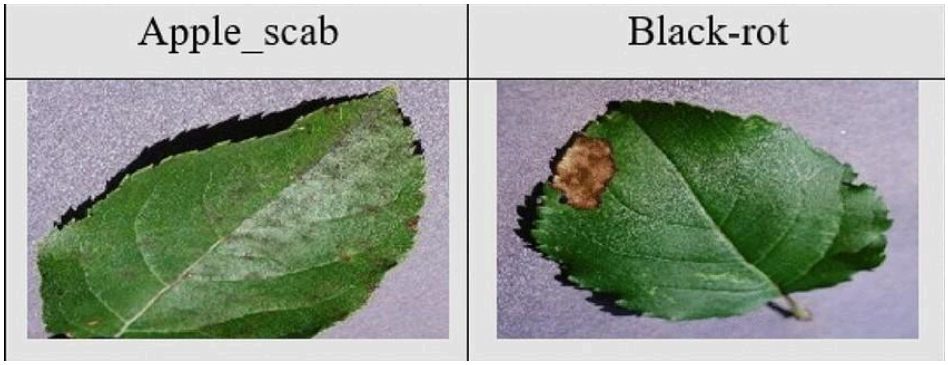


Figure 1: Apple scab and Black rot leaf [1]

Cedar apple rust is also a fungal infection caused by Gymnosporangium juniperi virginianae fungus. Initially, the reddish or pale yellow circular lesions appear on leaves' upper surface and gradually enlarge into bright orange yellow spots. Severe infection results in pale yellow or orange spots on fruits, and an unseasonable fall of leaves [5].

2. RELATED WORK

Several researchers have explored the application of deep learning and machine learning techniques for plant disease detection, particularly in apples and tomatoes.

Focusing on tomato plants, Mokhtar *et al.* [6] utilized Support Vector Machines (SVM) for tomato leaf disease detection. Their methodology, based on handcrafted features, represented an early machine learning approach prior to the widespread adoption of deep learning. Despite its simplicity, the method achieved satisfactory classification accuracy.

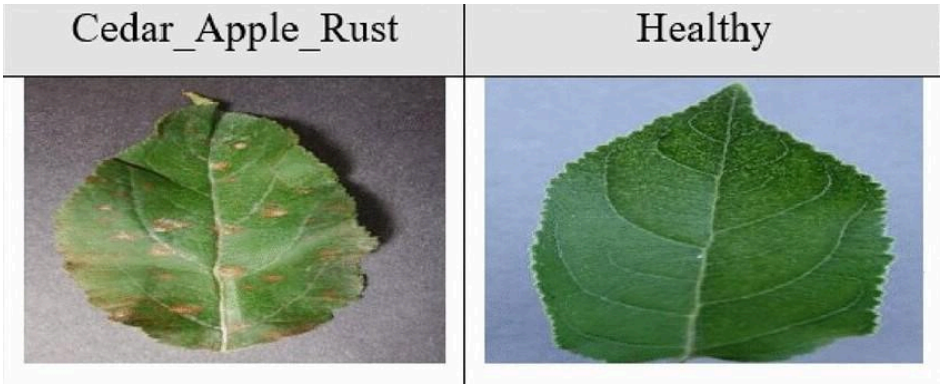


Figure 2: Cedar Apple Rust and Healthy leaf [1]

The steps involved in solving machine learning problems often include data collection, preprocessing, feature extraction, model training, and evaluation.

One of the key advantages of using deep learning techniques is the elimination of manual feature extraction. LeCun *et al.* [16] emphasized this strength of deep learning in their seminal work, noting that deep neural networks, particularly convolutional neural networks (CNNs), are capable of learning multiple levels of abstraction. These hierarchies enable the model to capture complex patterns and structures in data, making deep learning especially powerful in tasks such as image recognition, speech processing, and natural language understanding. Their study further established deep learning as a foundational approach for modern AI systems by demonstrating its superior performance in various domains without explicit feature engineering.

Tian *et al.* [7] proposed a multi-scale dense classification network for diagnosing typical apple diseases. Their approach achieved high accuracy and robustness by leveraging hierarchical features, which improved classification performance for complex disease symptoms.

Rafay *et al.* [2] introduced a real-time apple disease detection and classification system using a hybrid convolutional neural network (CNN) model was optimized for deployment in practical settings, offering a balance between speed and accuracy. The hybrid model demonstrated superior performance compared to traditional CNNs, especially in distinguishing visually similar diseases [9]. In a broader context, Schmidhuber provided a comprehensive overview of deep learning developments, including CNNs and their applications in image classification [8]. His work laid the theoretical foundation for many modern approaches to disease detection in plants and other visual tasks.

Furthermore, Brahimi et al. applied deep learning for the classification of tomato disease and the visualization of symptoms. Their work not only demonstrated effective disease classification but also emphasized explainability by visualizing affected regions, making

their system more in-terpretable for agricultural experts [9]. Northern maize leaf blight is a major disease threatening maize health. Detecting it is challenging due to complex field backgrounds and varying light conditions. He et al. introduced the ResNet architecture to address the degradation problem in deep neural networks. Instead of learning the desired mapping directly, ResNet uses residual connections to learn the residual functions, which significantly ease the training of deeper models [17]. These identity-based skip connections help preserve gradient flow across many layers, mitigating vanishing gradient issues and enabling the successful training of very deep networks like ResNet-50 and ResNet-101. Their model achieved state-of-the-art results on the ImageNet dataset and has since influenced various areas including object detection and segmentation.

Collectively, these studies highlight the evolution from classical machine learning to deep learning for plant disease detection, showcasing improved accuracy, efficiency, and usability in real-world scenarios.

3. DATASET

In this section, the origin of the dataset, the total number of images, and their distribution across various disease categories are described in detail.

A. Dataset Collection

In this study, the dataset used for apple leaf disease classification was obtained from Kaggle [20]. The dataset contains a total of 9,696 images of apple leaves, which include both diseased and healthy samples. The dataset is categorized into the following four classes:

Table 1: Distribution of Training and Test Images for Apple Leaf Classes

Class Name	Training Images	Test Images
Black Rot	1987	497
Cedar Apple Rust	1760	440
Healthy	2008	502
Apple Scab	2016	504

Each image in the dataset is a colored photograph of an apple leaf taken under various lighting and background conditions, simulating real-world scenarios. The high-quality and well-labeled images make this dataset suitable for training and evaluating deep learning models.

4. METHODOLOGY

Here, the dataset that is collected from internet is prepared, such that it can be used to pass in our CNN model. Following are the steps taken to build our model.

A. Data Preprocessing

Proper data pre-processing is needed to maintain uniform input. The following procedures were performed on the dataset:

a. Image Resizing And Normalization

All images were resized to a standard size of 224 × 224 pixels. The pixel values of the images initially range between 0 and 255. The values were normalized to between [0, 1] by dividing by 255.

b. Data Augmentation

In order to enhance model generalization and minimize overfitting, following data augmentation methods were applied:

- Rotation: Rotations of up to 20% of 360 degrees applied to simulate variability in leaf orientation.
- Flipping: Images are randomly flipped both horizontally and vertically.
- Zooming: Random zoom-in and zoom-out operations with a range of 20% are used to mimic the effect of varying camera distances.
- Shearing / Translation: Small translations are applied in width and height up to 10% of the dimensions of the image.

(a) Original (b) Zooming (c) Rotation (d) Shearing

Figure 3: Image transformations: original, zooming, rotation, and shearing [20]

B. Model Architecture

For the classification of apple leaf diseases, a custom Convolutional Neural Network (CNN) architecture was developed. Usually, a deep CNN is constructed by stacking several building blocks such as convolutional layers with a typical non-linear activation unit, pooling layers, and fully connected layers. etc. [16].

C. Convolutional Layer

The convolutional layer is responsible for performing convolution of a filter (kernel) on an input image. Fundamentally, the convolutional layer is a combination of two components: a linear convolution operation and a non-linear activation unit.

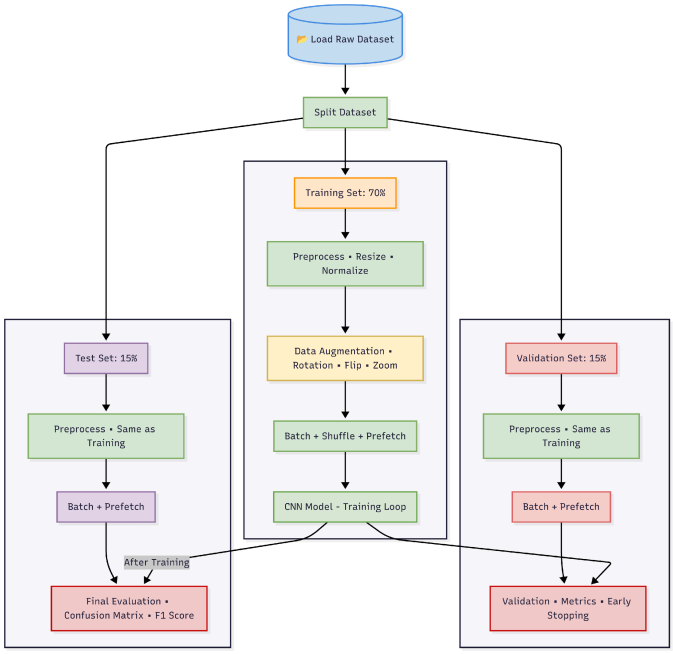


Figure 4: Data pipeline for apple disease classification

The convolution operation is performed over volumes of images with more than one channel (e.g., RGB images [10]) and is mathematically expressed as:

$$\text{conv}(I, K)(x, y) = \sum_{i=1}^{n_H} \sum_{j=1}^{n_W} \sum_{k=1}^{n_C} K_{i,j,k} \cdot I_{x+i-1, y+j-1, k} \qquad (1)$$

where the kernel $K \in R^{f_H \times f_W \times n_C}$ is convolved with an input image $I \in R^{n_H \times n_W \times n_C}$ having the same number of channels n_C to channels n_C, to generate a feature map $O \in R^{o_H \times o_W \times z}$. Here, f_H and f_W represent the height and width of the kernel, while n_H and n_W denote the height and width of the input image.

The dimension of the generated feature map is[10] defined as:

$$o_H = \left\lfloor \frac{n_H + 2p - f}{s} \right\rfloor + 1, \quad o_W = \left\lfloor \frac{n_W + 2p - f}{s} \right\rfloor + 1 \qquad (2)$$

where p denotes the padding size, s is the stride, and z is the number of filters used (i.e., number of feature maps generated). models [11]. Unlike other functions, ReLU does not activate all neurons simultaneously. The ReLU function is mathematically defined as follows

$$\text{ReLU}(x) = \max(0, x) \qquad (3)$$

D. Pooling Layer

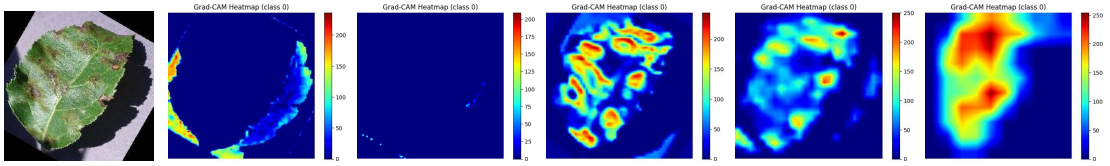
The pooling layer does down sampling of the feature maps that are generated by convolutional layers. The max pooling operation is represented mathematically as

$$y_j = \max_{i \in R_j}(P_i) \tag{4}$$

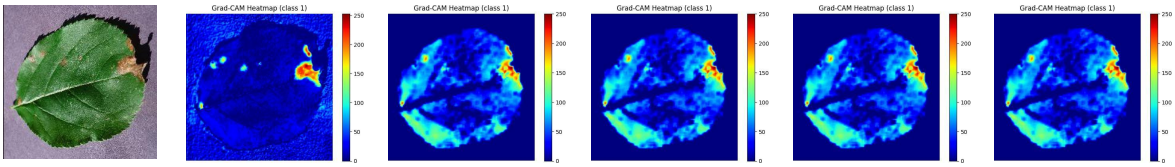
where R_j represents the receptive field that has pixel values P_i, and y_j represents the output of the pooling operation over that region.

The size of the feature map after pooling is calculated using [10]

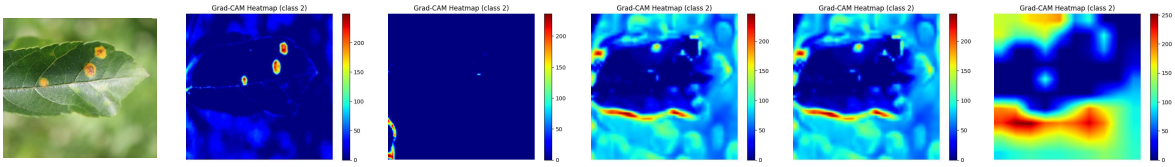
$$o_H = \left\lfloor \frac{n_H + 2p - f}{s} \right\rfloor + 1, \quad o_W = \left\lfloor \frac{n_W + 2p - f}{s} \right\rfloor + 1, \quad n_C = n_C \quad c \tag{5}$$



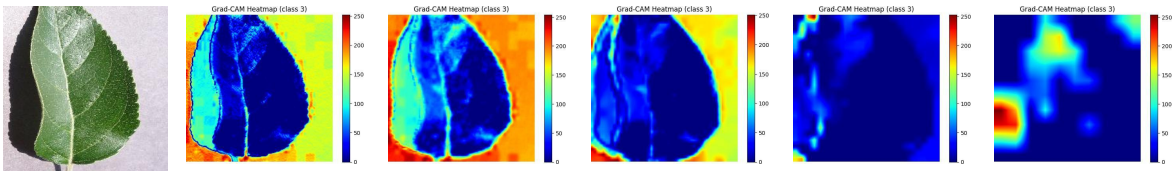
(a) Apple Scab (b) Conv1 (c) Conv2 (d) Conv3 (e) Conv4 (f) Conv5



(g) Black rot (h) Conv1 (i) Conv2 (j) Conv3 (k) Conv4 (l) Conv5



(m) Cedar rust (n) Conv1 (o) Conv2 (p) Conv3 (q) Conv4 (r) Conv5



(s) Healthy (t) Conv1 (u) Conv2 (v) Conv3 (w) Conv4 (x) Conv5

Figure 5: Grad-CAM visualizations of Apple leaf diseases. Each row shows one leaf type across convolution layers [4].

E. Fully Connected Layer

The fully connected (FC) layer, also called a dense layer, is identical to those used in conventional feedforward neural networks. FC layers are usually placed at the later stages of a CNN architecture and are utilized to generate output with the required number of classes or regression targets. Fully connected layer has two significant steps: a linear transformation followed by non-linear activation. These can be mathematically described as [18].:

$$Z = W^T X + b, O = f(Z) \tag{6}$$

Softmax function is used in the output section of the neural networks. This function is used normally in multi-class classification.

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{7}$$

F. Optimizer

An optimizer is an algorithm that adjusts the model's parameters (weights) based on the gradients of the loss function. [19].

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \\ \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t} \\ \theta_t &= \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \end{aligned} \tag{8}$$

Here, g_t is the gradient at time step t, m_t and v_t are the first and second moment estimates, β_1 and β_2 are the decay rates (commonly 0.9 and 0.999), α is the learning rate, and ϵ is a small constant (e.g., 10^{-8}) to prevent division by zero.

G. Loss Function

Categorical cross-entropy is a commonly used loss for multi-class classification problems, especially with softmax output layers [18].

$$\mathcal{L}_{\text{CCE}} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{9}$$

Here, $C = 4$ is the number of classes, y_i is the true class label, $\hat{y}_i$ is the predicted probability for class i from the softmax output.

H. Gradient Weight Class Activation Mapping

In order to understand our convolutional neural network (CNN) model, Gradient Weighted Class Activation Mapping (Grad CAM) is used as explainability technique. [2] [12]. Grad-CAM generates a class discriminative heatmap that highlights important regions in the input image influencing the model's decision.

Given a target class c, and a convolutional layer with feature maps A^k, Grad-CAM computes the importance weights α^c for each feature map by global average pooling the gradients:

$$\alpha_k^c = \frac{1}{Z}\sum_i\sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{10}$$

where y^c is the score for class c (before softmax), and Z is the total number of pixels in the feature map.

The class-specific localization map $L^c_{Grad-CAM}$ is then obtained as:

$$L^c_{\text{Grad-CAM}} = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right) \tag{11}$$

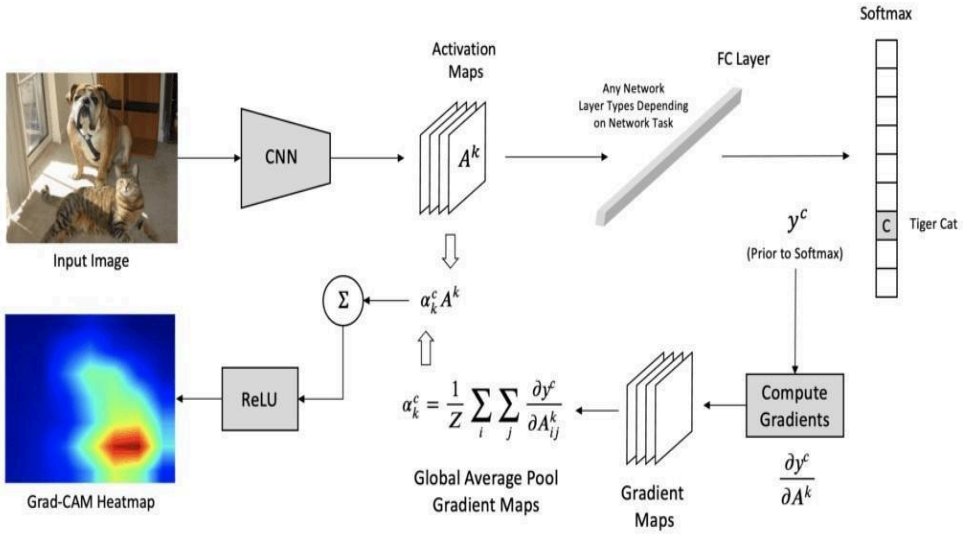


Figure 6: Grad-CAM visualization
[Image Source: https://xai-tutorials. readthedocs.io]

Activation map highlighting the most influential regions in the input image.

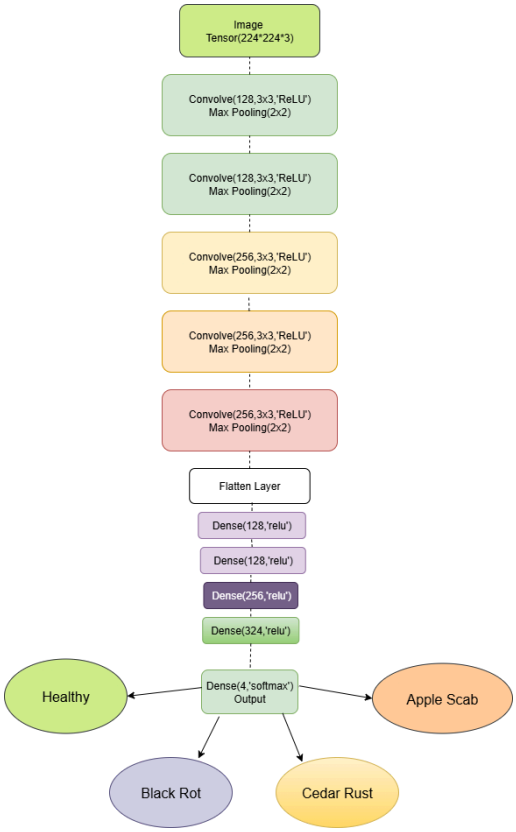


Figure 7: CNN architecture with convolution, pooling, and fully connected layers

Table 2: CNN Model Summary

Layer Type	Layer Name	Output Shape	Kernel / Stride	Parameters
Input	–	(224, 224, 3)	–	0
Conv2D	conv1	(222, 222, 128)	3×3 / 1×1	3,584
MaxPool2D	pool1	(111, 111, 128)	2×2 / 2×2	0
Conv2D	conv2	(109, 109, 128)	3×3 / 1×1	147,584
MaxPool2D	pool2	(54, 54, 128)	2×2 / 2×2	0
Conv2D	conv3	(52, 52, 256)	3×3 / 1×1	295,168
MaxPool2D	pool3	(26, 26, 256)	2×2 / 2×2	0
Conv2D	conv4	(24, 24, 256)	3×3 / 1×1	590,080
MaxPool2D	pool4	(12, 12, 256)	2×2 / 2×2	0
Conv2D	conv5	(10, 10, 256)	3×3 / 1×1	590,080

MaxPool2D	pool5	(5, 5, 256)	2×2 / 2×2	0
Flatten	flatten	(6400)	–	0
Dense	dense1	(128)	–	819,328
Dense	dense2	(128)	–	16,512
Dense	dense3	(256)	–	33,024
Dense	dense4	(324)	–	83,148
Dropout	dropout	(324)	–	0
Dense	output	(4)	–	1,300

5. RESULTS AND DISCUSSION

This part starts with a brief description of the experimental setup used for the research, followed by an explanation of different performance metrics. Grad-CAM was applied for visualization and and compare the outputs of convolutional layers, which gave insight into which regions of the input image is responsible for CNN's predictions.

A. Experimental Setup

The proposed Convolutional Neural Network (CNN) model was developed and trained using Google Colab, a cloud based development environment that provides access to free GPU;s. The implementation was done in Python 3, using key libraries such as TensorFlow and Keras to build and train deep learning models. The input images were resized to 224×224 pixels, and the dataset was split into training and testing sets to evaluate the model's performance. Data augmentation techniques, including rotation, flipping, zooming, and translation, were applied to enhance gener-alizability of the model.

B. Performance Metrics

To evaluate the effectiveness of the CNN model for apple leaf disease classification, several standard performance metrics were used[3].

- **Accuracy:** The ratio of correctly predicted observations to the total observations.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

- **Precision:** The ratio of correctly predicted positive observations to the total predicted positives, indicating the model's exactness.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

- **Recall (Sensitivity):** The ratio of correctly predicted positive observations to all actual positives, measuring the model's completeness.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

- **F1-Score:** The weighted average of Precision and Recall, especially useful for imbalanced datasets.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

- **Specificity:** The proportion of true negatives among all actual negatives. It reflects the model's ability to correctly identify healthy leaves or exclude a disease when it's not present.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{16}$$

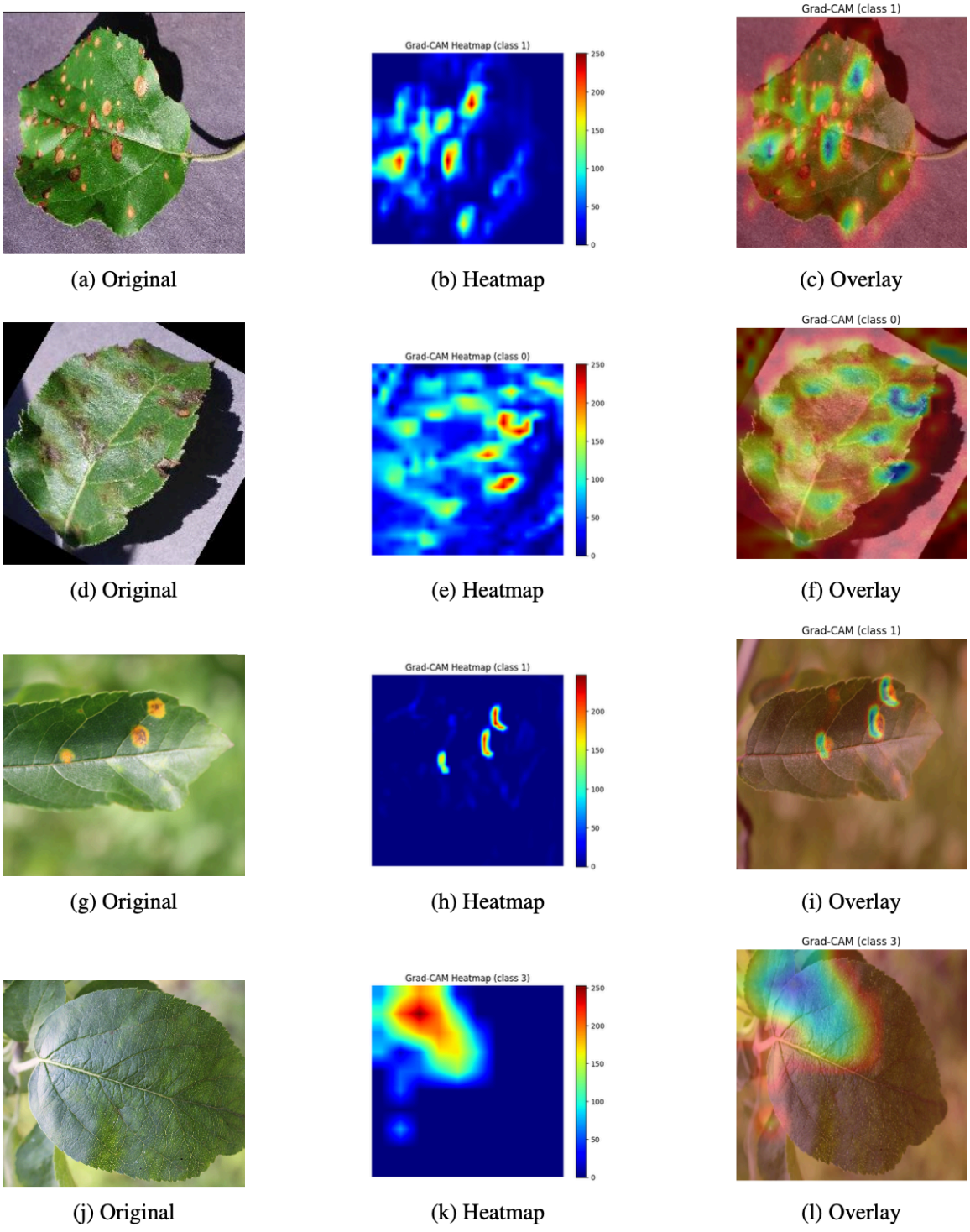
- **Confusion Matrix:** A visual representation of prediction results, showing true vs. predicted classifications for each class.

These metrics are calculated to get a comprehensive assessment of the model's performance in terms of both correctness and robustness across all four classes of disease classification.

C. Model Interpretability Using Grad-CAM

To better understand the model's predictions, we applied Grad CAM to visualize class-discriminative regions for correctly and incorrectly classified images. Figure 8 illustrates examples of original images, their Grad-CAM heatmaps, and overlay results.

The attention of convolution layer was focused mostly on the regions where the diseases were found in correctly classified leaves, It validated the relevance of learned features.



(a) Original (b) Heatmap (c) Overlay

(d) Original (e) Heatmap (f) Overlay

(g) Original (h) Heatmap (i) Overlay

(j) Original (k) Heatmap (l) Overlay

Figure 8: Illustrates examples of original images, their Grad-CAM heatmaps, and overlay results. [4]

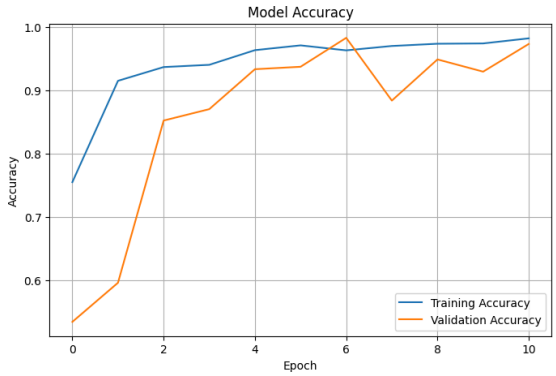


Figure 9: Training Accuracy vs Validation Accuracy.

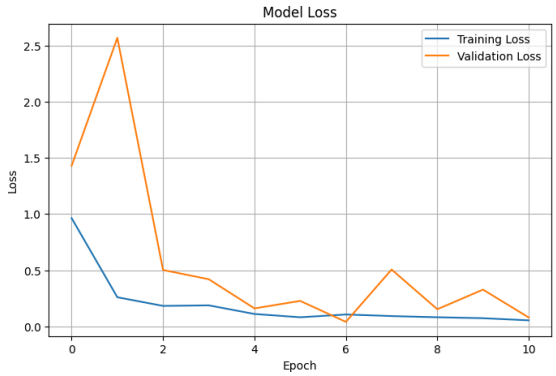


Figure 10: Training Loss vs Validation Loss.

As shown in the graph. At first both validation and training accuracy are low, with the increase in epochs the model improved it's accuracy and reduced the loss. Validation accuracy of the model at the end reached 97.36%.

Table 3: Classification Report with Specificity

Class	**Precision**	**Recall**	**Specificity**	**F1-score**	**Support**
Apple scab	0.98	0.96	0.994	0.97	374
Healthy	0.96	0.99	0.984	0.97	414
Cedar apple rust	0.99	0.99	0.996	0.99	367
Black rot	0.97	0.95	0.990	0.96	399
Macro avg	0.97	0.97	0.991	0.97	1554
Weighted avg	0.97	0.97	0.991	0.97	1554
Accuracy	0.9736				

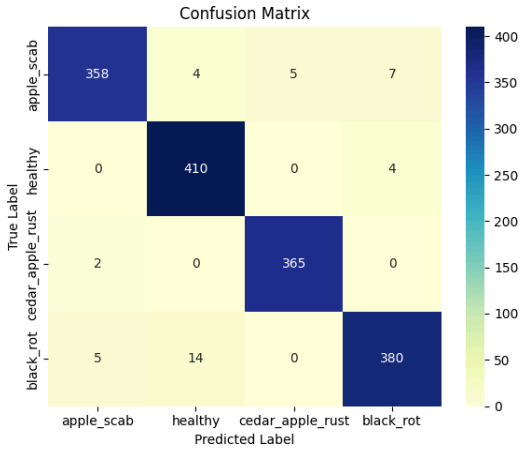


Figure 11: Confusion Matrix for Apple Disease Classification

6. CONCLUSION

The suggested convolutional neural network (CNN) model had good performance in the classification of four given apple leaf diseases with an overall accuracy of 97.36% Precision, recall, F1-score, and specificity values were all high for all classes, revealing well-balanced and consistent detection capability. The model performs quite better at detection apple scab and cedar rust.

A. Limitation

The model only detects four apple diseases and does not cover others or pest damage. Limited dataset diversity may affect generalization. Also, this model still struggles to detect the diseases on leaf where the background differ drastically as in the figure 8 we can see that heatmap face difficulty to exactly point out the disease spot. Since apple leaves can have multiple diseases at the same time, so model may not be able to clearly distinguish the disease exactly.

B. Recommendation

Future efforts must be directed towards the enrichment of the dataset with varied samples and try investigating sophisticated models such as ResNet or EfficientNet for better performance. Including more environmental information can help improve the forecasting of apple disease.

ACKNOWLEDGMENT

I would like to thank the Department of Electronics and Computer Engineering, Thapathali Campus, for providing the necessary academic environment. I am particularly grateful to honorable lecturer, Er. Rajad Shakya, for his precious guidance and encouragement during the study. I am grateful to all the authors and researchers whose work has been quoted and used as a basis for this project.

REFERENCES

[1] P. Bansal, R. Kumar, and S. Kumar, "Disease Detection in Apple Leaves Using Deep Convolutional Neural Network," *Agriculture*, vol. 11, no. 7, p. 617, 2021.

[2] A. Rafay, M. Aqeel, M. Iqbal, A. Sohaib, B. Islam, and A. Zaheer, "Real-Time Apple Disease Detection and Classification Using Hybrid CNN Model," *International Journal of Advanced Natural Sciences and Engineering Researches*, vol. 7, no. 10, pp. 251–259, 2023.

[3] A. Ge´ron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., O'Reilly Media, 2019.

[4] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[5] D. M. Kabala, A. Hafiane, L. Bobelin, and R. Canals, "Image-based crop disease detection with federated learning," *Scientific Reports*, vol. 19220, Nov. 2023, doi: 10.1038/s41598-023-46218-5.

[6] U. Mokhtar, M. A. Ali, A. E. Hassenian, and H. Hefny, "Tomato Leaves Diseases Detection Approach Based on Support Vector Machines," in *Proc. 2015 11th Int. Computer Engineering Conf. (ICENCO)*, pp. 246–250, 2015, doi: 10.1109/ICENCO.2015.7416336.

[7] Y. Tian, E. Li, Z. Liang, M. Tan, and X. He, "Diagnosis of typical apple diseases: a deep learning method based on multi-scale dense classification network," *Frontiers in Plant Science*, vol. 12, p. 698474, 2021.

[8] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61,

pp. 85–117, 2015.

[9] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep Learning for Tomato Diseases: Classification and Symptoms Visualization," Preprint, 2017.

[10] S. Theodoridis, "Chapter 18 - Neural Networks and Deep Learning," in *Machine Learning*, 2nd ed., Academic Press, New York, NY, USA, 2020, pp. 901–1038, doi: 10.1016/B978-0-12-818803-3.00030-1.

[11] Z. Dong, X. Chen, W. Jia, S. Du, K. Muhammad, and S.-H. Wang, "Image-based fruit category classification by 13-layer deep convolutional neural network and data augmen-tation," *Multimedia Tools and Applications*, vol. 78, no. 3, pp. 3613–3632, 2019, doi: 10.1007/s11042-017-5243-3.

[12] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," in *Proc. IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626, 2017.

[13] Z. Iqbal, M. Khan, M. Sharif, and J. Shah, "An automated detection and classification of citrus plant diseases using image processing techniques: A review," *Computers and Electronics in Agriculture*, vol. 153, pp. 12–32, Aug. 2018,

doi: 10.1016/j.compag.2018.07.032.

[14] A. Badage, “Crop disease detection using machine learning: Indian agriculture,” *International Research Journal of Engineering and Technology*, vol. 5, pp. 866–869, 2018.

[15] J. Canny, "A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, pp. 679–698, 1986, doi: 10.1109/TPAMI.1986.4767851.

[16] Y. LeCun, Y. Bengio, and G. Hinton, “Deep Learning,” *Nature*, vol. 521, pp. 436–444, 2015, doi: 10.1038/nature14539.

[17] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in

Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA,

pp. 770–778, June 2016.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, 2016.

[19] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[20] Ludehsar, *Apple Disease Dataset*, [Online]. Available: https://www.kaggle.com/datasets/ludehsar/apple-disease-dataset. Accessed: July 2, 2025.