2023 26th International Conference on Computer and Information Technology (ICCIT)
13-15 December, Cox's Bazar, Bangladesh

# RoseVision: An Android Application To Detect Rose Leaf Diseases Using Modified Convolutional Neural Network

Saidul Islamm Nayan[1], Syed Mahfuzur Rahman[2], Nosin Ibna Mahbub[3]
[1,2]*Dept. of Computer Science and Engineering, University of Global Village, Barishal-8200, Bangladesh*
[3]*Dept. of Information and Communication Technology, Islamic University, Kushtia-7003, Bangladesh*
Email- nayan.saidul@gmail.com, mahfuzrahman129@gmail.com, mahbub.nosin@gmail.com

***Abstract*—Every year 28% of the global cut rose production is hampered by various rose plant diseases. Early detection and classification could diminish the cost of cultivation lost. However, without proper knowledge and expert assistance, farmers are facing challenges to identify the diseases in time to take action. Previously, a few works have been conducted on this. So far, no study has been done to develop an efficient automatic mobile application detection system with faster, lighter, and more accurate predictions. In this study, we presented an artificially intelligent mobile application using image processing to identify rose leaf diseases. The primary goal of this work is to create a simple but exact identification with minimal computing power and time. We proposed a Modified Convolutional Neural Network (MCNN) to classify different rose leaf diseases. The proposed MCNN model outperforms the famous pre-trained models including VGG16, VGG19, DenseNet121, MobileNetV2, Resnet50, InceptionV3, and Extreme Inception (Xception) by achieving 97.94% test accuracy. Additionally, based on the trained MCNN model, we developed an android-based mobile application that will allow the farmers to quickly identify the disease using their android phones.**

***Index Terms*—Rose Leaf Disease, Agriculture, Deep Learning, CNN, Disease Identification, Android Application.**

## I. Introduction

Among 369,000 species of flowering plants, the rose is the prettiest, sweetest, and most beautiful creation of nature. Their beauty, fragrance, and appearance refresh our minds and souls, which is why they are called the king of flowers. The rose belongs to the family Rosaceae. They come in different colors, such as white, red, yellow, pink, etc. Different colors symbolize different emotions, like love, beauty, happiness, and compassion. Rose oil, water, and fragrance are also used in the cosmetic industry. So people of different ages, casts, and areas use roses for different purposes. To meet this increasing demand, rose cultivation has increased in different countries globally. The global annual sale of cut roses is around 1 billion US dollars [1]. The commercial cultivation, quality, and agricultural production of roses are decreasing due to different diseases of rose leaves. So, early detection of the diseases that affect various parts of rose leaves is crucial for large-scale production. Diseases caused by bacteria, fungi, algae, and microorganisms that damage the natural growth of plants show up on the plant's leaf, flower, fruit, or body. The traditional way to detect these diseases is visually by human experts. However, farmers don't have enough knowledge to identify the disease on their own. Various automatic leaf disease detection systems have been enhanced through the application of machine learning techniques to address this issue. The recent improvement in machine learning has opened new doors for image classification, but most of them use larger and deeper convolutional neural network (CNN) models that take more time to train and test. Here, we designed a modified convolutional neural network (MCNN) which is a lightweight CNN model that takes less time to train and test correctly. We applied our model to the publicly available dataset FlowerNet [2]. The entire size of the dataset is 4342 augmented images of three different classes: black spot, downy mildew, and disease-free fresh rose leaves. Our MCNN model accuracy is better than most of the recognized CNN models out there. To compare and prove our statement, we ran different famous pre-trained models on the same dataset as well. Finally, we created a mobile application that will allow users to take pictures and identify the correct disease class in real-time. This improved model accuracy and automatic disease identification system will help the farmers increase the quality and production of roses.

Several works have already been done on plant disease, specifically rose leaf disease detection [3] [4]. Some papers focused on different image pre-processing techniques and classifier functions, whereas others used different CNN models. To predict the classes more accurately, a properly modified neural network and a robust dataset are required. In [5], researchers have applied a faster R-CNN algorithm to detect multiple disease classes in different plants. The main objective was to check the feasibility of plant disease identification and also propose an automatic system. Their obtained accuracy was 67.34%. Authors in [6] detected different rose plant diseases using the MobileNet model, incorporating and excluding transfer learning approaches. They showed that MobileNet with a transfer learning approach has a better accuracy of 95.63 percent than without transfer learning of about 85.73%. In [7], authors offered an optimal neural network model for plant disease detection and recognition implemented on a mobile platform. A two-step training: pre-training on ImageNet, and then post-training on particular plant illnesses has been used. The test accuracy for the model was 89.0%. Various segmentation and feature extraction techniques have been employed to identify the leaf's damaged section in [8]. This work emphasized area-based and edge-based segmentation techniques. The statistical classification-based neural network classifier described in [9] could correctly detect and categorize the evaluated illnesses with a precision score of 93%. In the first step, they used K-means image segmentation, followed by a pre-trained neural network. The study's [10] findings point to improved performance, with a disease classification accuracy rate of 97.3%. They augmented their images using generative adversarial networks (GANs) and applied two different CNN models, Inception v3 and MobleNet in this paper [11]. Inception V3 scored 88.6% and 92% accuracy with the MobileNet model. In [12], a hybrid deep learning model derived from the VGG16 architecture outperforms applying early fusion techniques and SVM classifiers rather than using the softmax function. After using 10-fold cross-validation, the accuracy was 90.26%. The researchers in [13], worked on CNN with different layers and found that the 5-layer model with 15 epochs scored the highest validation accuracy of 95.05%.

After reviewing earlier studies, we found that there is no encouraging research that can assist farmers in rapidly identifying rose leaf disease. To address this deficiency, we propose RoseVision, an Android application with a faster and more accurate prediction that allows farmers to effortlessly identify rose leaf diseases and take appropriate action.

## II. Materials and Methods

The main aim of this research is to develop a lightweight but still more accurate CNN model to predict rose leaf diseases. After getting the best performing model, we integrated it into a mobile application for easy accessibility. The overall plan of this study is broken up into two blocks. The first layer consists of data collection, preparation, and model training. And the top-performing trained model was used to create an Android app in the next layer. Every step of this research is displayed in **Figure 1**.

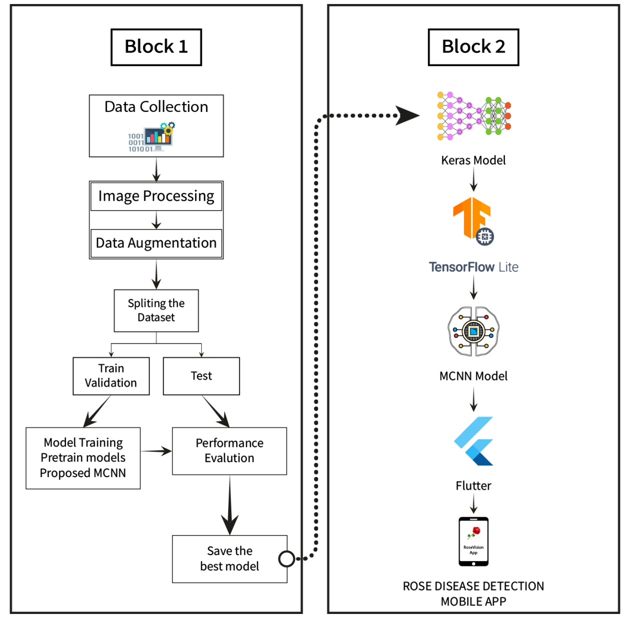



Fig. 1: The entire procedure of this study

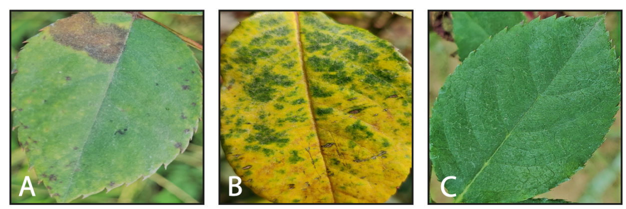



Fig. 2: Example of rose leaf diseases: A) Black Spot B) Downy Mildew C) Fresh Leaf

### Layer A: Data Processing and Model Building

### *A. Compiling and pre-processing data*

This study makes use of the RoseNet dataset [2]. It is an extensive dataset of 4342 images of rose leaves, both diseased and disease-free, classified into three classes. **Table I** enumerates every category and the quantity of photos in every class. **Figure 2** represents a selection of images from the dataset [2].

TABLE I: Collected dataset with number of images per classes

| Name of the category | Count of images |
|---|---|
| Black Spot | 1434 |
| Downy Mildew | 1478 |
| Fresh Leaf | 1430 |

Data Preprocessing is the first step of preparing it for deep learning model training. All the acquired images from the dataset were gone through a set of steps: (i) Image Acquisition (ii) Preprocessing the images by reducing their resolution from their original size (iii) The Images then standardized, allowing the loss function to obtain the global optimum as rapidly as possible. (iv) Next, data augmentation was carried out to reduce data overfitting and boost accuracy [14].

### *B. Proposed Modified Convolutional Neural Network (MCNN)*

**Figure 3** depicts the full visual architecture of our proposed MCNN model. The resolution of the input images was 224x224 pixels and normalized (RGB) color channels before being supplied to the model. Our suggested MCNN architecture is as follows:

1) The beginning layer starts with a 3x3 pixels kernel and 32 filters. ReLU activation occurs after the earlier process. Following is the max-pooling layer, which has a stride of 1 pixel and a kernel size of 2x2. Getting 32x111x111 feature maps at the end.
2) The subsequent layer has a 3x3 pixel kernel and 64 filters. ReLU activation occurs after the earlier process. Following is the max-pooling layer, which has a stride of 1 pixel and a kernel size of 2x2. Getting 64x54x54 feature maps at the end.

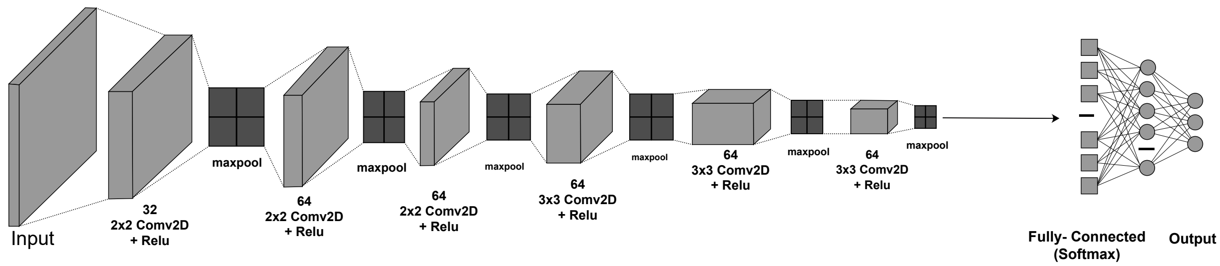



Fig. 3: Architecture of the proposed MCNN model

3) Thirdly, 3x3 pixel kernel size of 64 filters with ReLU activation, 2x2 pixel max-pooling layer, and a stride of 1 pixel resulting in 64x26x26 feature maps.
4) 64 filters of 3x3 pixel kernel and ReLU activation make up the fourth convolution layer. These filters are followed by max-pooling with a 2x2 pixel kernel size and a 1-pixel stride to create feature maps of 64x12x12.
5) The fifth convolution layer consists of 64 filters including 3x3 kernel and ReLU activation, then max-pooling of 2x2 pixel kernel and a 1-pixel stride, producing 64x5x5 feature maps.
6) Finally, it involves 64 filters with a 3x3 pixel kernel, ReLU activation, and max-pooling with a 2x2 pixel kernel size and a 1-pixel stride, producing 64x1x1 feature maps.
7) Following that, the first fully connected layer consists of 64 neurons from flattening the last convolutional layer outputs and is accompanied by a ReLU activation function. The fully connected layer then includes three output neurons with a softmax function that finally produces a distribution probability for three output classes.

We used six convolutional layers and one dense layer. We reduced the image's dimension by using max-pooling layers, and relu being a function of activation to provide nonlinearity to the network. The dataset was split by 70% into train, 20% into validation, and 10% into test sets. We used 35 epochs with a batch size of 32 and an Adam optimizer for each CNN model. The parameters used to train all the models are shown in **Table II**.

TABLE II: The parameters used for all the models

| Parameter Name | Values |
|---|---|
| Number of epochs | 35 |
| Batch size | 32 |
| Loss function | Categorical cross entropy |
| Learning rate | 0.001 |
| Optimizer | adam |
| Train: Validation: Test | 70:20:10 |

## Layer B: Mobile Application Development

### C. Concept Design

After training the MCNN model, we created the RoseVision smartphone application for the Android operating system, allowing farmers to directly profit from this research. **Figure 4** illustrates the conceptual structure utilized in order to develop this app. With the RoseVision application, the user could capture a rose plant leaf and diagnose its ailment. The developed RoseVision app will process and evaluate the captured image using the MCNN algorithm to identify the plant disease. Cross Entropy, Train Accuracy, and Validation Accuracy were calculated using pixels from the sample image or photograph. The output will contain the results of the analysis performed with the trained model. The results will show whether the plant is healthy or has any ailments. If a disease has been identified on the rose plant, the app will also provide specific disease details.

### D. System Architecture and Development

In order to build the RoseVision mobile app, we initially saved the trained MCNN Keras model as a h5 extension. After that, the Keras model was converted to a Tensorflow lite model. TensorFlow's portable version for smartphones and embedded devices is known as TF-Lite. It primarily makes use of machine learning models created specifically for each activity to provide greater speed and accessibility on portable devices with lower latency. It enables effective local execution of TF models on end devices. It offers a high API level that understands the model's output and changes the incoming data [22]. Finally, we used Flutter to create the RoseVision app utilizing the translated tflite model [23]. **Figure 1** shows a simplified visual representation of the entire procedure. The RoseVision application's system architecture is shown in **Figure 6**. It demonstrates how users, programs, and devices communicate with one another. The design suggests that the hardware, an Android-based smartphone, uses a camera to detect leaf disease, which is the app's main component. It also demonstrates the existence of two disease detection modules that allow users to identify the rose leaf disease. The camera unit is used as a tool to photograph the plant's leaf.

TABLE III: Evaluates how well modern CNN architectures perform across different categories and presents the top five error rates.

| Architecture Name | Year | Number of Parameters | Error rate | Depth | Reference |
|---|---|---|---|---|---|
| AlexNet | 2012 | 60 M | ImageNet: 16.4 | 8 | [15] |
| VGG | 2014 | 138 M | ImageNet: 7.3 | 19 | [16] |
| ResNet | 2016 | 25.6 M | ImageNet: 3.6 | 152 | [17] |
| InceptionV3 | 2016 | 23.8 M | ImageNet: 3.5 | 48 | [18] |
| MobileNetV2 | 2018 | 3.4 M | ImageNet: 1.7 | 19 | [19] |
| Xception | 2017 | 22.8 M | ImageNet: 0.055 | 126 | [20] |
| DenseNet | 2017 | 25.6 M | CIFAR-10+: 3.46 | 190 | [21] |

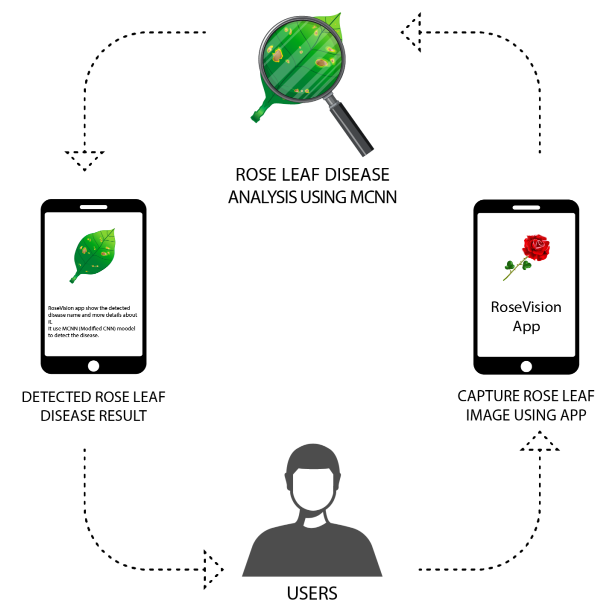



Fig. 4: Conceptual architecture of app development

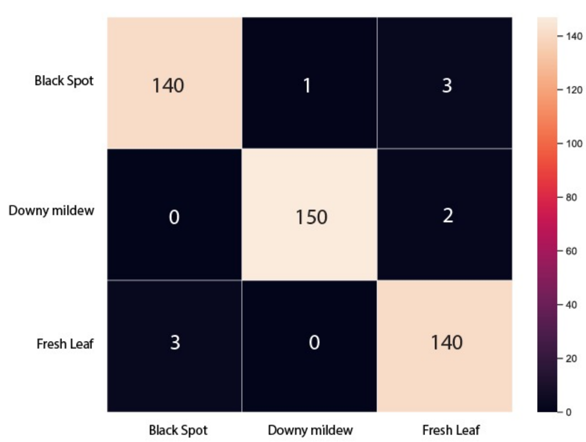



Fig. 5: Proposed MCNN model's confusion matrix

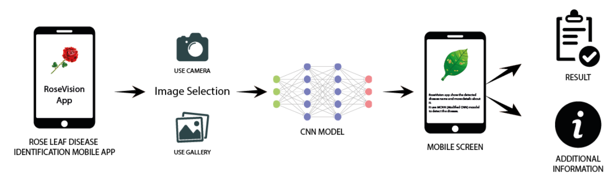



Fig. 6: System architecture of RoseVision app

Additionally, the user has the option of selecting a saved leaf picture from the phone gallery.

## III. Experimental results

The proposed model takes images as input to classify several rose leaf diseases. For better training and testing, we split the pre-processed dataset into train, validation, and test into 70%, 20%, and 10% accordingly. To compare with our MCNN model we used the same dataset for different recognized transfer learning models. The performance of recent CNN architectures used in this study is represented in **Table III**. Here, MobileNetv2 is the lightest model having only 3.4 million parameters with a depth of 19 whereas VGG19 from the VGG family has the most number of parameters with the same layer depth. In contrast, MobileNetV2 and Xception obtained the highest accuracy. So, to develop a new lightweight but still more accurate architecture than these we worked on a modified CNN and performed training and testing on the preprocessed dataset. After training and testing, model evaluation and performance analysis is the next most important task. There are various ways to evaluate a model. We considered four standard metrics: Precision (P), Recall (R), Accuracy (A), $f_1$-score ($f_1$). **Table IV** represents a detailed comparison between the proposed MCNN and different pre-trained models.

### A. *Experimental Result Analysis*

The main focus was on the model effectiveness over compared transfer learning models and the performance of our developed app on test images. The MCNN model obtained 99.60% training accuracy and 98.43% validation accuracy while testing accuracy of 97.94%. There is a 6.30% and 0.68% loss in both validation and training respectively. **Figure 7**

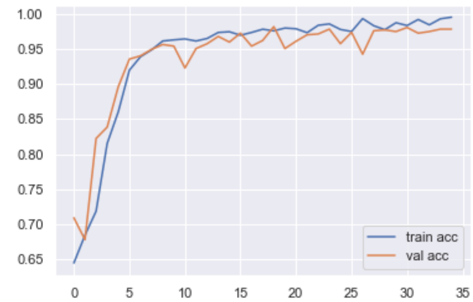
TABLE IV: For each category, a comparison of the F1 scores, recall, and precision of various models is made

| Model | Class Name | Precision | Recall | F1-score |
|---|---|---|---|---|
| DenseNet121 | Black Spot | 0.95 | 0.99 | 0.97 |
| | Downy Mildew | 0.99 | 0.99 | 0.99 |
| | Fresh Leaf | 0.99 | 0.96 | 0.98 |
| MobileNetV2 | Black Spot | 0.97 | 0.98 | 0.97 |
| | Downy Mildew | 0.99 | 0.99 | 0.99 |
| | Fresh Leaf | 0.97 | 0.96 | 0.97 |
| InceptionV3 | Black Spot | 0.93 | 0.90 | 0.92 |
| | Downy Mildew | 0.98 | 0.99 | 0.99 |
| | Fresh Leaf | 0.92 | 0.94 | 0.93 |
| ResNet50 | Black Spot | 0.68 | 0.61 | 0.64 |
| | Downy Mildew | 0.92 | 0.94 | 0.93 |
| | Fresh Leaf | 0.70 | 0.76 | 0.73 |
| VGG16 | Black Spot | 0.94 | 0.97 | 0.95 |
| | Downy Mildew | 0.99 | 0.98 | 0.98 |
| | Fresh Leaf | 0.96 | 0.94 | 0.95 |
| VGG19 | Black Spot | 0.99 | 0.86 | 0.92 |
| | Downy Mildew | 0.98 | 0.98 | 0.98 |
| | Fresh Leaf | 0.88 | 1.0 | 0.94 |
| Xception | Black Spot | 0.93 | 0.98 | 0.96 |
| | Downy Mildew | 0.99 | 0.99 | 0.99 |
| | Fresh Leaf | 0.98 | 0.92 | 0.95 |
| Proposed MCNN | Black Spot | 0.98 | 0.97 | 0.98 |
| | Downy Mildew | 0.99 | 0.99 | 0.99 |
| | Fresh Leaf | 0.97 | 0.98 | 0.97 |

represents the loss and accuracy of training and validation. **Figure 5** shows the confusion matrix of our MCNN model. We also compared our model with some existing pre-trained models including VGG16, VGG19, DenseNet121, MobileNetV2, Resnet50, InceptionV3, and Xception for further evaluation. From **Figure 8** we can see that the ResNet50 obtained the lowest of 76.94% while our MCNN scored the highest at 97.94% in terms of test accuracy. DenseNet121 and MobileNetV2 scored the same accuracy of 97.71%. VGG16 and Xception obtained slightly above 96% while VGG19 and InceptionV3 scored above 94%. In the final step, we tested and evaluated our app with both normal and affected images of various rose leaf diseases. We tested our MCNN along with all the pretrained CNN models on 438 completely unknown images and calculated the average testing time. The **Figure 9** shows the average testing time of all the trained models. Here, our proposed MCNN took the lowest time of 0.074 second while VGG19 took the highest testing time. **Figure 10** displays some illustrations of how the app can effectively recognize various rose leaf diseases and the final predictive results interface of the app.

## IV. Conclusion and Future Work

Globally, the agriculture sector has embraced artificial intelligence (AI) by implementing smart edge technologies to boost productivity and decrease losses due to struggling with increasing inflation, declining mineral richness, and harsher regulations. The health and quantity of crops farmed for flowers and other cultivable goods are substantially impacted by plant diseases. Without enough specialized aid, farmers




(a) Accuracy throughout each training epoch


(b) Per epoch loss in training phase

Fig. 7: Curve of performance for proposed MCNN model

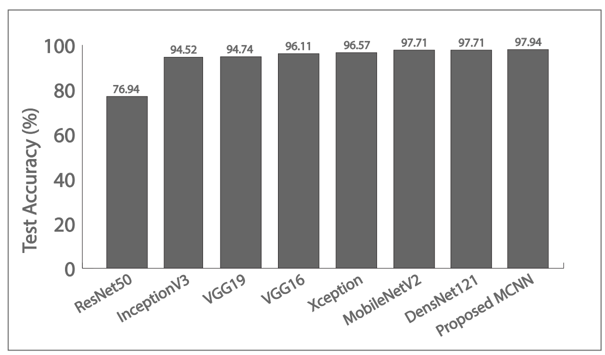



Fig. 8: Comparison of test accuracy among all the trained models

generally spend a substantial amount of money combating diseases of rose, which has detrimental effects. Unfortunately, the variety or characterization of rose diseases affecting roses in Bangladesh is mostly unknown. Thus, accurate disease detection has become an essential aspect of modern agriculture. In this paper, a Modified Convolutional Neural Network (MCNN) is offered to distinguish between healthy leaves and two different rose leaf diseases (Black Spot and Downy Mildew). With a testing accuracy of over 97%, our provided MCNN model surpassed the pretrain CNN architecture, which is a substantial improvement. Ultimately, through using the trained MCNN model, we developed the ”RoseVision” Android app,

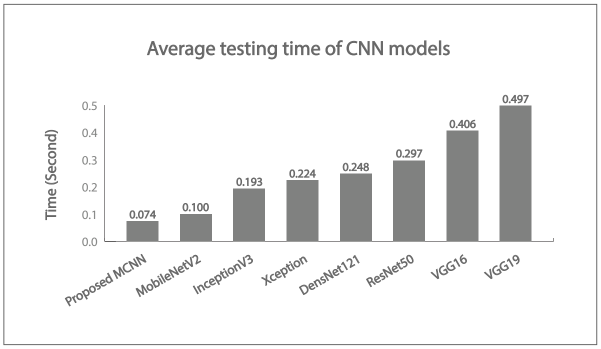



Fig. 9: Average testing time of trained CNN models

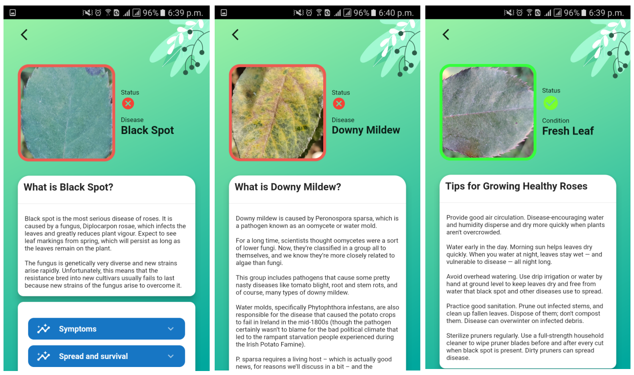



Fig. 10: Correctly identified samples black spot, downy mildew, fresh leaf from different rose leaf disease images.

allowing farmers to promptly identify the disease and take the appropriate steps to cure it. The system will be trained in the future for newer leaf disease classification. Additionally, a suggestion technique will be created to give the farmer a detailed strategy on how to treat that illness. Some of our upcoming objectives are:

- Working in collaboration with plantations to create a more diverse dataset.
- Tweaking the MCNN model's parameters, such as the number of layers, hidden nodes, and activation function.
- Improving the process of feature extraction.
- Developing an effective disease-tracking framework and deploying it on an automated drone could be advantageous for inspecting the area and collecting attributes to determine the crop's true state of health.

## References

[1] M. Sudha, "Economics of protected cultivation of cut-rose for international markets: Role of price prediction models in integrating production with marketing," *Agricultural Economics Research Review*, vol. 14, no. 1, pp. 1–15, 2001.

[2] S. Sazzad, A. Rajbongshi, R. Shakil, B. Akter, and M. S. Kaiser, "Rosenet: Rose leave dataset for the development of an automation system to recognize the diseases of rose," *Data in Brief*, vol. 44, p. 108497, 2022.

[3] S. H. Khan, M. N. Hossain, S. Mozumder, S. Sharmin, and N. N. Moon, "Rose leaf disease detection using cnn," in *2021 IEEE International Conference on Robotics, Automation, Artificial-Intelligence and Internet-of-Things (RAAICON)*. IEEE, 2021, pp. 37–41.

[4] M. I. Khaleel, P. G. Sai, A. U. R. Kumar, P. Raja, and V. T. Hoang, "Rose plant leaves: Disease detection and pesticide management using cnn," in *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2022, pp. 1067–1072.

[5] S. T. Cynthia, K. M. S. Hossain, M. N. Hasan, M. Asaduzzaman, and A. K. Das, "Automated detection of plant diseases using image processing and faster r-cnn algorithm," in *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*. IEEE, 2019, pp. 1–5.

[6] A. Rajbongshi, T. Sarker, M. M. Ahamad, and M. M. Rahman, "Rose diseases recognition using mobilenet," in *2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. IEEE, 2020, pp. 1–7.

[7] A. L. P. De Ocampo and E. P. Dadios, "Mobile platform implementation of lightweight neural network model for plant disease detection and recognition," in *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*. IEEE, 2018, pp. 1–4.

[8] S. D. Khirade and A. Patil, "Plant disease detection using image processing," in *2015 International conference on computing communication control and automation*. IEEE, 2015, pp. 768–771.

[9] D. Al Bashish, M. Braik, and S. Bani-Ahmad, "A framework for detection and classification of plant leaf and stem diseases," in *2010 international conference on signal and image processing*. IEEE, 2010, pp. 113–118.

[10] K. Swetharani and G. V. Prasad, "Design and implementation of an efficient rose leaf disease detection and classification using convolutional neural network," *International Journal of Image Mining*, vol. 4, no. 1, pp. 98–113, 2021.

[11] R. Gandhi, S. Nimbalkar, N. Yelamanchili, and S. Ponkshe, "Plant disease detection using cnns and gans as an augmentative approach," in *2018 IEEE International Conference on Innovative Research and Development (ICIRD)*. IEEE, 2018, pp. 1–5.

[12] S. Nuanmeesri, "A hybrid deep learning and optimized machine learning approach for rose leaf disease classification," *Engineering, Technology & Applied Science Research*, vol. 11, no. 5, pp. 7678–7683, 2021.

[13] P. K. Kosamkar, V. Kulkarni, K. Mantri, S. Rudrawar, S. Salmpuria, and N. Gadekar, "Leaf disease detection and recommendation of pesticides using convolution neural network," in *2018 fourth international conference on computing communication control and automation (ICCUBEA)*. IEEE, 2018, pp. 1–4.

[14] E. Ayan and H. M. Ünver, "Data augmentation importance for classification of skin lesions via deep learning," in *2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT)*. IEEE, 2018, pp. 1–4.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[20] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[21] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[22] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang *et al.*, "Tensorflow lite micro: Embedded machine learning for tinyml systems," *Proceedings of Machine Learning and Systems*, vol. 3, pp. 800–811, 2021.

[23] N. Kuzmin, K. Ignatiev, and D. Grafov, "Experience of developing a mobile application using flutter," in *Information Science and Applications*. Springer, 2020, pp. 571–575.